Computerized Cooking (1985)

Computerized Cooking

David A. Mundie

Culinary Softwary Systems

Pittsburgh.

Cataloguing in Publication Data

Mundie, David.
Computerized Cooking.

1\. Recipes, theory of. I. Title.

641.01

Table of Contents

Introduction

In our times, recipes have become unusable. Alternately enshrined behind plexiglas bookstands in glossy, expensive, full-color extravaganzas, and consigned to the never-never land of journalistic evanescence, they have become a form of literature more than a help to the practicing cook. Julia Child and her co-conspirators, objective allies of Burger King and Wendy's, shroud good cooking in mysticism and ritual, making it seem out of reach of us mortals, while our day-to-day eating continues to degenerate in spite of good intentions and a sincere rebirth in gastronomic interests. Fast foods and fad foods prosper, while the sense of culinary traditions, of cooking as an ongoing art form, wanes perilously. Overwhelmed by the absurd burden placed on them by the peculiar notion that each published recipe must be "original," recipe writers make gratuitous, undocumented modifications to recipes just to avoid lawsuits, and the notion of authenticity goes out the window.

To restore the recipe to its rightful place as a technical document to facilitate the labors of the practicing cook in this era where there is simply no time to cook, two things are needed. First, a notation must be adopted which captures the essence of recipes more clearly, more cleanly, and more succinctly than the prosaic recipes that are the norm today. Secondly, the enormous power offered by microcomputers must be harnessed to give instantaneous access to the totality of the world's cuisines, and to aid the cook in the onerous tasks of recipe selection, shopping list generation, and recipe analysis.

The formal recipe language RxOL, designed at Culinary Software Systems, offers help on both these fronts. First, it is a revolutionary new approach to simplified cooking. Using it one can easily and quickly take in a new recipe, identify its structure, and execute it in a customized fashion with a minimum of anxiety and in the shortest time possible. Second, because RxOL is a formal language, it readily lends itself to computer analysis, with all the benefits which that entails.

The first part of this work, A Prolegomenon to Recipology, examines the notational and computational issues from a theoretical perspective. Chapter 1 presents a critique of current recipe styles and offers an axiological perspective on what recipes ought to achieve. Chapter 2 summarizes the benefits that one may reasonably expect computerized cooking to offer.

The second half of the book turns to the practical aspects of computerized cooking. Chapter 3 presents the RxOL notation that lies at the heart of the Cocina software offered by Culinary Software Systems, and shows how it meets the criteria established in Chapter 1. Chapter 4 is an introduction to the Cocina software itself, and surveys the state of of the implementation of the tools described in Chapter 2.

Chapter 1. A Rationale for Recipes

In this section we are concerned with the advantages RxOL offers to humans. In order to present the superiority of RxOL over other types of recipes, we must first address ourselves to the question, "What is a recipe?" We begin by presenting a set of recipe attributes which can be used to form a taxonomy of recipe styles. Using this taxonomy, we proceed to examine a sample of different recipe styles from around the world, to deepen our understanding of the way recipes function as texts. With this background, we offer a critique of prevailing recipe styles and argue that they are totally unsuited to the heavy burdens placed upon them by contemporary society.

What is a Recipe?

Before the Industrial Revolution, recipes were relatively unproblematic. Serving primarily as crib sheets for experienced cooks, they encapsulated the basic ingredients and manipulations needed to put a dish on the table as succinctly and simply as possible. In fact the ingredients which enter into a dish and the manipulations which are applied to those ingredients are the sine qua non, the degree zero, of recipes.

With the societal changes which placed more and more culinary responsibility on individuals—housewives, husbands—who were not professional cooks, recipes began to absorb into themselves vast quantities of extraneous information in an attempt to make the recipes seem more accessible. In our age recipes ramble on about what Spanish fisherman eat at lunch, about how good Aunt Millie's kitchen smelled when the peach cobbler was baking, about the splendid sights and sounds and fresh ingredients provided by an Arab souk. So swamped have recipes become, it seems advisable to catalogue the ingredients of recipes as they appear in the modern world:

> **Sociology and history.** In an attempt to make recipes more interesting, modern authors have often incorporated into their works general comments about, for example, the history of the Phillipines, the culinary regions of Italy, cultural influences in Indonesia, the indomitable spirit of the Armenians. This is great stuff. Anyone seriously interested in food and cooking must enjoy such reading. The only issue is whether it belongs in recipes; I argue that it does not.

> **Travelogue.** The line between sociology and travelogue can be a fine one. Nevertheless, it seems useful to distinguish a general commentary on the mores of a country from what one man ate at that fabulous cafe in Lisbon as the sun sank slowly in the west.

> **Chitchat.** The line between travelogue and chitchat can also be difficult to discern. There is a history-sociology-travelogue-chitchat continuum, and the cuts one makes in it are always somewhat arbitrary. I classify as chitchat any information, such as what Señor Gonzales was wearing when he served that superb Paella, or how many letters irate readers sent poor James Beard when he published a recipe calling for only 2 3/4 cups of flour in a 1-2-3-4 Cake, which says nothing about the raison d'être of the recipe at hand.

> **Culinary technique.** Doubtless due in part to the lower average level of expertise on the part of their audience, modern cookbook authors have all too often succumbed to the temptation to encumber their recipes with detailed descriptions of how the operations involved are to be executed. My favorite example is the ritualistic incantation at the start of so many stir-fried recipes: "The oil should be hot enough to cook with when the first tiny bubbles form and a few small wisps of smoke appear." A critical examination of almost any popular cookbook will reveal an endless supply of superfluous, insulting drivel that Shannon would characterize as having no information content. "Serve." (As opposed to what? throwing it in the garbage?) "Seed the green pepper." "Wash the tomatoes." "Peel the garlic." "Use a sharp knife."

> **Culinary equipment.** Paradoxically, information on the proper utensils for a recipe is an intrusion on the recipe itself. The fact of the matter is that noodles cooked in a coffee pot taste just fine, that steak sautéed in a pressure cooker, and potatoes boiled in a frying pan, and omelettes made in a wok, and cake baked in a souflé dish, etc., etc., are virtually indistinguishable from their more usual equivalents. The choice of equipment is largely a matter of common sense and the material at hand. A cook who, reading "sauté the onion in butter," needs to be told to do it in a frypan, needs a general education, not a more verbose recipe.

> **Biology.** Another realm of knowledge which impinges on cooking, and which therefore cookbook authors have felt compelled to include in recipes, is biology, particularly in the forms of agriculture, animal husbandry, and piscatology. To be fair, such information is often relegated to a "dictionary of ingredients" at the back of the book, but its intrusion into recipes per say is persistent enough to warrant mention here. Descriptions of Mediterranean fish in a Bouillabaisse recipe, of exotic fruits in a kiwi recipe, of Galangale in an Indonesian curry recipe, are all too familiar.

> **Measurements.** The experienced cook, with a developed flair for how flavors combine, has little use for measurements. With the exception of certain baked goods, the combinations involved in cooking are physical, not chemical; errors in measurement will affect the taste of the dish, but not its fundamental makeup. As with equipment, the fact of the matter is that wide variations in quantities are quite acceptable. It does not really matter whether your beef stew has 200 g of potatoes, or 500, or 1000.

> Modern recipe authors, however, in their misguided attempt to substitute precise details for intuitions, have raised measurements to shibboleth status. It was years before I realized that "kitchen tested" means that someone has prepared the dish using exactly the measurements given, as though measuring ingredients to six decimal places would somehow ensure the cook's success. Nothing saddens me more than the sight of a cook scrupulously measuring out 1/4 teaspoon of thyme for a dish using one of the "measuring spoon" sets that are endemic in the Anglo-Saxon kitchen.

> Nevertheless, it seems reasonable for a recipe to give some indication of what quantities are involved. What is unreasonable is for the cook to feel obligated to follow those indications slavishly. This does not argue, however, for the sloppy, informal style of measurements, which specify a "glass" of water, a "large" onion, and so forth; paradoxically, such imprecision clouds the cook's responsibility for the measurement in a way that "200 ml" does not. Measurements in cooking should be precise, but not necessarily accurate.

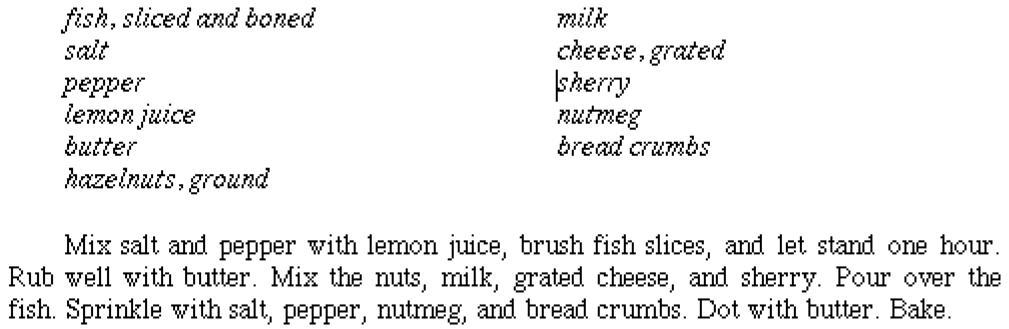
> **Nutritional Information.** With the rise of concern over the effects of food on health, it was inevitable that calorie counts and so on would invade recipes. From the point of view of the practitioner trying to get the meal on the table, however, they are just one more impediment. Being calculable from the recipe itself, there is no need for them in a computerized cooking environment.

> Photographs. For completeness, I conclude with the recipial intrusion I find least objectionable, namely photographs of the completed dish. Although they do tend to distract the cook by overspecifying the appearance of the dish (I have known people to rush around in vain attempts to duplicate a table setting seen in Gourmet), they can be delightful to look at and art objects in their own right.

* * *

What is a recipe? It is the thesis of this work that a recipe is none of the above, but rather an expression in the mathematical sense, that is, a sequence of operands and operators which yield a result which is the dish whose preparation is being described. The rules for generating such sequences define the deep structure of recipes, just as grammatical rules define the deep structure of sentences, phonetic rules define the deep structure of utterances, and the anthropological rules of Lévi-Strauss define the deep structure of myths. In fact, it is in the last analysis the conflict between the syntactic rules of natural language and those of cooking which makes natural language an unsuitable vehicle for describing recipes. In computer science terminology, recipe languages should not be imperative, but rather functional.

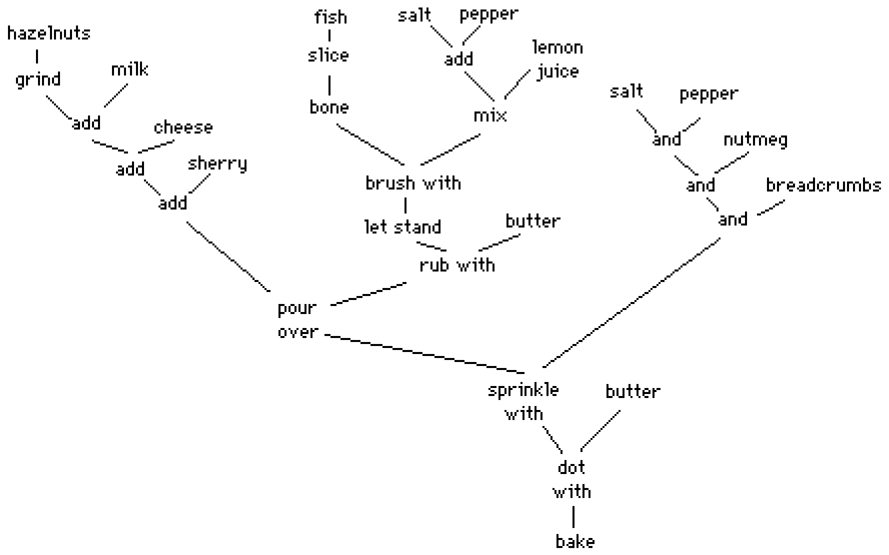
Consider for example the following recipe, written in American Standard Style, with details such as measurements omitted:



fish, sliced and boned
salt
pepper
lemon juice
butter
hazelnuts, ground
milk
cheese, grated
sherry
nutmeg
bread crumbs

Mix salt and pepper with lemon juice, brush fish slices, and let stand one hour. Rub well with butter. Mix the nuts, milk, grated cheese, and sherry. Pour over the fish. Sprinkle with salt, pepper, nutmeg, and bread crumbs. Dot with butter. Bake.

The deep structure of this recipe, as defined by the sequence of operands and operators which make it up, is shown by the following tree diagram:



Such tree structures have been extensively studied in computer science, and their properties are well known. They are made up of three kinds of nodes: terminals, such as "fish", which are the operands or raw material of the tree; unary operators, such as "grind", which take one operand and yield one new operand (in our example, ground hazelnuts); and binary operators, such as "sprinkle with," which combine two operands to yield a new, compound operand (in our example, the sprinkled fish).

As we shall see from our survey of recipe styles in the next section, no traditional recipe notation has been designed to bring out the essential tree structure of recipes. All of them do violence to the deep structure, by forcing the recipe into a prose straightjacket or by splitting the tree into pieces. It is the desire to express precisely and succinctly the deep structure of recipes which led to the development of RxOL.

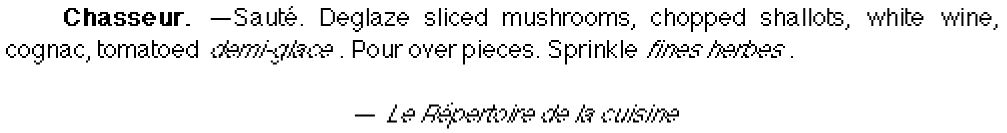
A Sampling of Recipe Styles.

Having examined the elements which make up recipes, let us examine some typical recipes to try to answer the question of what makes a recipe style good or bad.

Before beginning, let us consider how recipe styles might be classified. It seems clear that the inclusion or omission of each of the elements identified in the preceding section constitutes half of such a taxonomy. All recipes which include informal measurements, abundant travelogue, and photographs, share a family resemblance.

The second half of such a taxonomy takes into account the way the chosen elements are arranged. Is the travelogue information intertwined with the recipe itself, or is it rather set out in a clearly demarcated preliminary paragraph? Do the photographs accompany the recipes, or are they gathered into a special section by themselves? As we shall see, the evolution of the "Standard" recipe style is largely a matter of the separating of ingredients and the unary operations applied to them from the rest of the recipe.

Sample 1: Aide-Mémoire Style.



Chasseur. —Sauté. Deglaze sliced mushrooms, chopped shallots, white wine, cognac, tomatoed *demi-glace* . Pour over pieces. Sprinkle *fines herbes* .

— *Le Répertoire de la cuisine*

This elegantly concise style comes close to being Pure Recipe. It is almost a formal language in its own right; one feels that building an automatic parser for it would be fairly easy. Nothing that could slow down the actual preparation of the dish is included; no sociology, no chitchat, no measurements, no nothing. Even the name of the principle ingredient has been omitted, on the grounds that it can be inferred from the fact that this recipe is in the chicken section. It is assumed that the cook has elsewhere learned how to sauté and deglaze.

The result is that dozens of these recipes fit on a page. The eye can quickly scan large bodies of cuisine looking for just the right recipe for a given occasion. Because the overhead for printing each recipe is so small, the cookbook writer is encouraged to be more comprehensive in his coverage. When the recipe is ladened with great masses of unnecessary material, one is lucky to get 200 recipes in an entire cookbook; the Répertoire has that many in just a few pages.

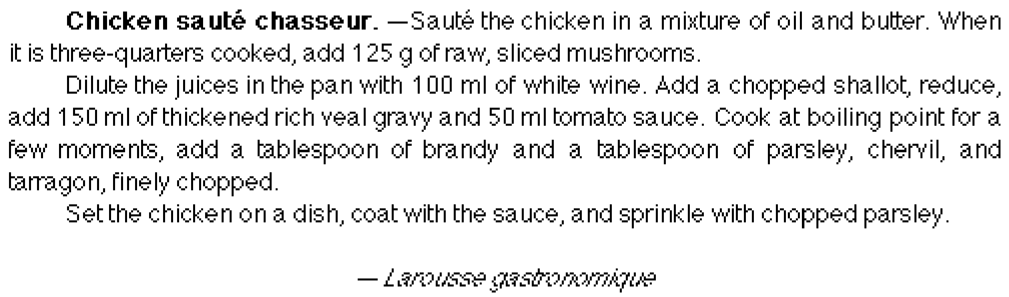
A corollary to this is that with longer recipes, the corpus becomes scattered over a large number of cookbooks. To find just the French recipe one is looking for, one might have to go through an entire shelf of ordinary cookbooks, or one could just look in the Répertoire.

In the preface to their work, the authors of the Répertoire offer one of the very few discussions of recipe style to be found anywhere. They themselves point out the correlation between brevity and comprehensive coverage: "All we wanted to do was to present in a restricted format the largest possible number of ancient and modern recipes." The small size and enormous coverage of their work allows working cooks always to have it "at hand", and always to find "the desired recipe in the heat of the moment."

They make the fundamental distinction between recipes as literature and recipes as tools: "our work, far from having any literary pretensions, frankly adopts a professional vocabulary which grammarians will perhaps find audacious, but which all cooks, even novices, will understand without effort." Moreover, they argue, as I have, that considerations of culinary technique have no place in recipes:

"Need we say that it is not in this pocket book that a cook should learn all the secrets, the refinements of his art, but rather in the great works of our masters?" Finally, they point out the importance of nomenclature for recipology: "Every day a well-intentioned chef baptizes with a new name a dish long known by another; every day, also, that a cook presents under an already "registered" name a preparation different from the one that name evokes; those are serious errors against which every chef conscious of his mission has the duty to protest with us; for such errors will necessarily lead culinary art towards decadence, in spite of all the science, in spite of all the efforts of our masters."

Sample 2: Larousse Style.



Chicken sauté chasseur. —Sauté the chicken in a mixture of oil and butter. When it is three-quarters cooked, add 125 g of raw, sliced mushrooms.

Dilute the juices in the pan with 100 ml of white wine. Add a chopped shallot, reduce, add 150 ml of thickened rich veal gravy and 50 ml tomato sauce. Cook at boiling point for a few moments, add a tablespoon of brandy and a tablespoon of parsley, chervil, and tarragon, finely chopped.

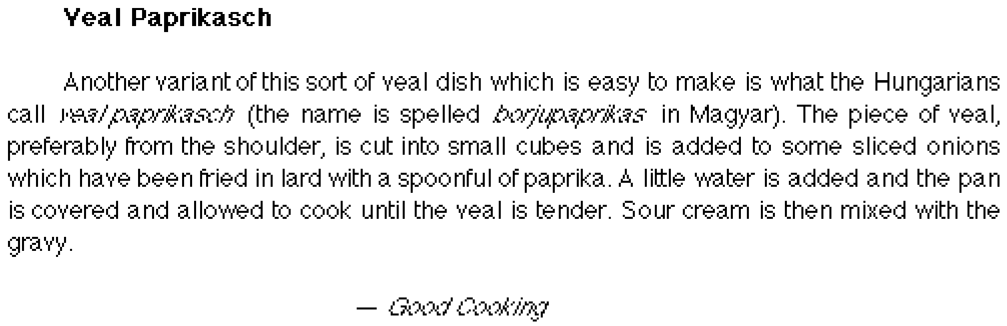
Set the chicken on a dish, coat with the sauce, and sprinkle with chopped parsley.

— *Larousse gastronomique*

This sample exemplifies the "literary" approach to recipes. Everything possible is done to make the recipe look like a normal prose passage. This is a lie, of course, for recipes are not prose. Consider the division of the recipe above into paragraphs. Nothing in the recipe itself corresponds to the division we are given; the paragraph breaks might just as well have come at "When" and "Cook", for example. The prose format is a straightjacket into which the recipe is forced willy-nilly.

The recipe is still relatively concise. A certain number of measurements, some formal, some informal, are provided. A couple of inane details have crept in ("Set the chicken on a dish, not on the floor") and some subroutines have been expanded in-line ("fines herbes" has become "parsley, chervil, and tarragon"), but most of the technical details have been left out.

Sample 3: Narrative Prose



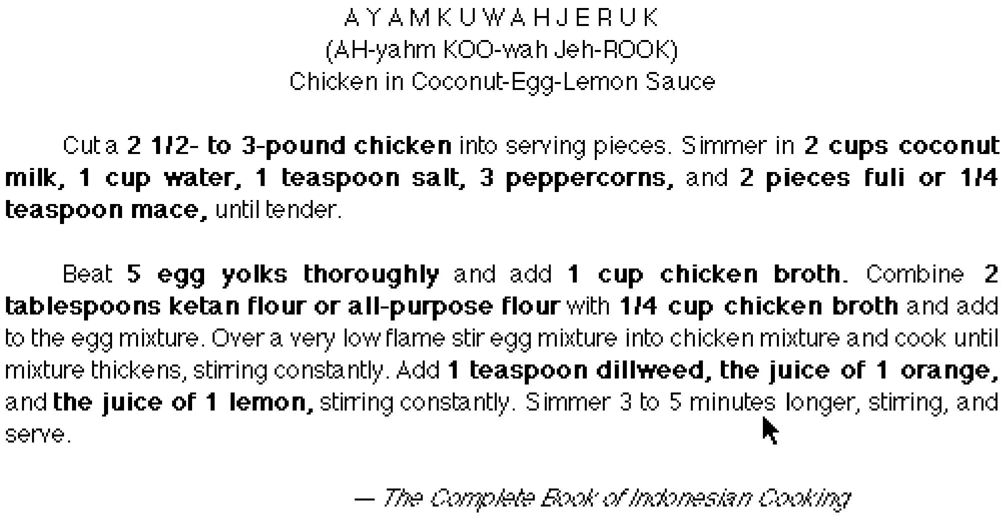
Veal Paprikasch

Another variant of this sort of veal dish which is easy to make is what the Hungarians call *veal paprikasch* (the name is spelled *borjupaprikas* in Magyar). The piece of veal, preferably from the shoulder, is cut into small cubes and is added to some sliced onions which have been fried in lard with a spoonful of paprika. A little water is added and the pan is covered and allowed to cook until the veal is tender. Sour cream is then mixed with the gravy.

— *Good Cooking*

This carries the "literary" style a step further. Instead of frankly giving instructions for preparing the dish, the author pretends to be describing the cultural habits of the Hungarians. For the sake of literary style, the natural chronology of the recipe is dismembered ("is cut into cubes and is added to some sliced onions which have been fried in lard"). The recipe is invaded by culinary taxonomy ("another variant"), by sociology ("what the Hungarians call"), and by chitchat ("the name is spelled") The omission of measurements is doubtless motivated by the desire to appear literary.

Sample 4: Emboldened Prose.



AYAMKUWAHJERUK
(AH-yahm KOO-wah Jeh-ROOK)
Chicken in Coconut-Egg-Lemon Sauce

Cut a **2 1/2- to 3-pound chicken** into serving pieces. Simmer in **2 cups coconut milk, 1 cup water, 1 teaspoon salt, 3 peppercorns,** and **2 pieces fuli or 1/4 teaspoon mace,** until tender.

Beat **5 egg yolks thoroughly** and add **1 cup chicken broth.** Combine **2 tablespoons ketan flour or all-purpose flour** with **1/4 cup chicken broth** and add to the egg mixture. Over a very low flame stir egg mixture into chicken mixture and cook until mixture thickens, stirring constantly. Add **1 teaspoon dillweed, the juice of 1 orange,** and **the juice of 1 lemon,** stirring constantly. Simmer 3 to 5 minutes longer, stirring, and serve.

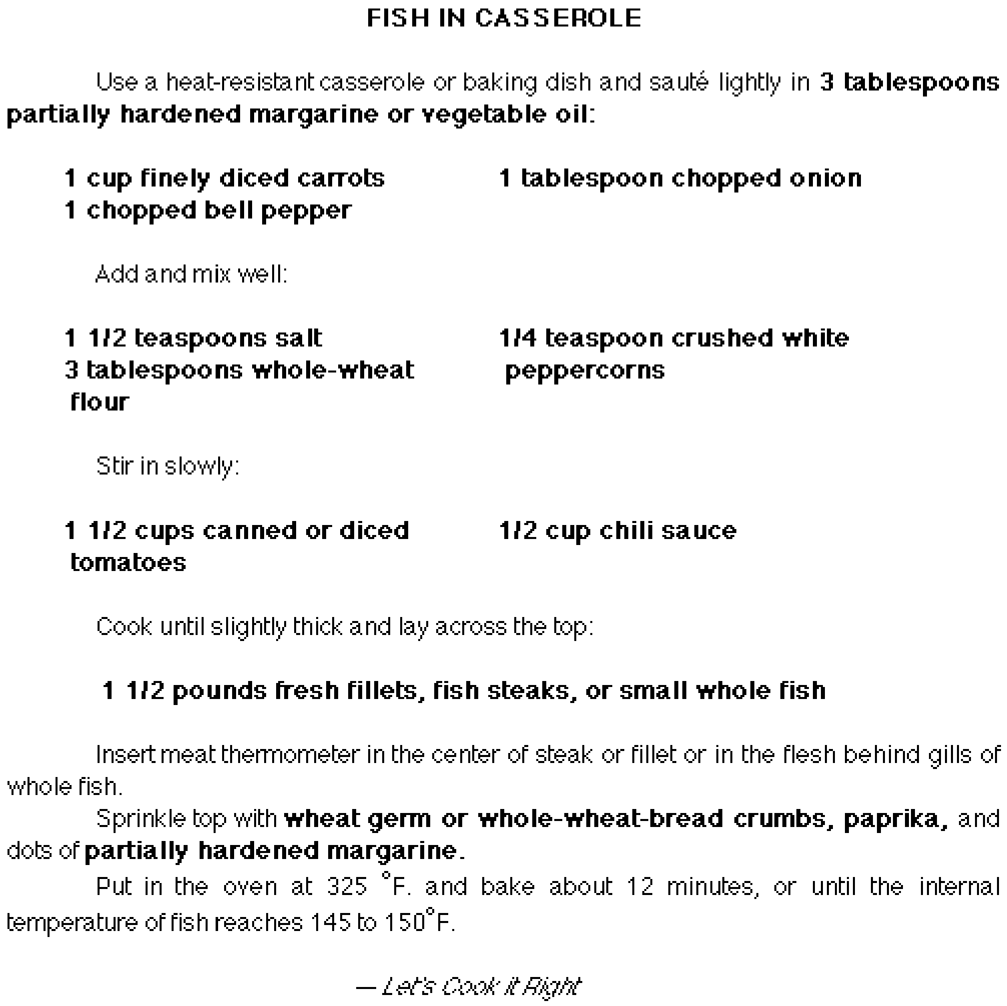
— *The Complete Book of Indonesian Cooking*

With this recipe we see the beginning of the Great Mitosis, the cultural change which resulted in Customary Style through the ripping off of the ingredients of a recipe, along with the measurements and operations which apply just to those ingredients taken one by one. Beginning not in a temporal sense (this is a relatively recent recipe) but in a conceptual one. This recipe is a very close cousin of the Larousse gastronomique style. Its random division into paragraphs, its occasional meaningless instructions ("serve"), its omission of anything but ingredients, operations, and measurements, all suggest the Larousse. It is true that the use of purely formal measurements gives it a less literary tone. But what really sets this recipe apart is the use of bold-face type for ingredients.

The motivation for this special treatment is presumably to allow the cook to make up a shopping list quickly. With software tools to generate the shopping list, such a motivation loses its force. There is however another, subtler motivation: the sense that it is the ingredients which give the recipe a structure, that it is important for the cook to be able to scan a recipe and take in its ingredient list quickly, to look ahead to see "what's coming up next."

Mitosis in this form is relatively harmless. It is relatively easy for the eye to ignore the bold-facing and read the recipe in a normal manner.

Sample 5: Health Food Imperative Style.



FISH IN CASSEROLE

Use a heat-resistant casserole or baking dish and sauté lightly in **3 tablespoons partially hardened margarine or vegetable oil:**

1 cup finely diced carrots
1 chopped bell pepper
1 tablespoon chopped onion

Add and mix well:

1 1/2 teaspoons salt
3 tablespoons whole-wheat flour
1/4 teaspoon crushed white peppercorns

Stir in slowly:

1 1/2 cups canned or diced tomatoes
1/2 cup chili sauce

Cook until slightly thick and lay across the top:

1 1/2 pounds fresh fillets, fish steaks, or small whole fish

Insert meat thermometer in the center of steak or fillet or in the flesh behind gills of whole fish.

Sprinkle top with **wheat germ or whole-wheat-bread crumbs, paprika,** and dots of **partially hardened margarine.**

Put in the oven at 325 °F. and bake about 12 minutes, or until the internal temperature of fish reaches 145 to 150°F.

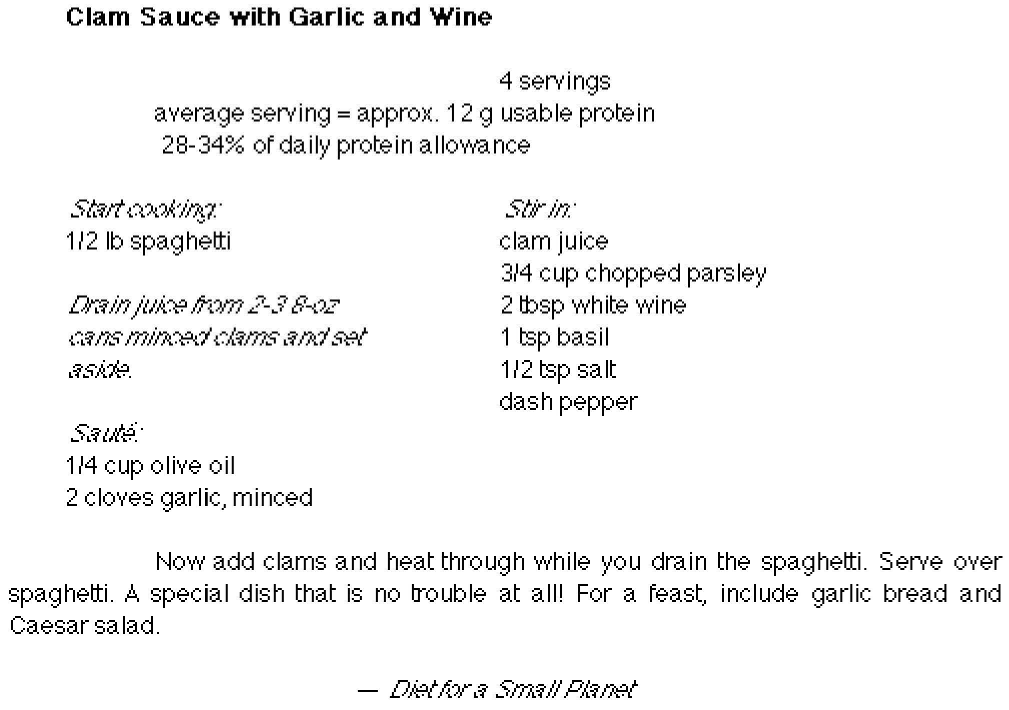
— *Let's Cook it Right*

Here the Mitosis has progressed one step further. As in the preceding case, ingredients and their measurements have been boldfaced, but in addition some of them have been separated from the text and surrounded by white space.

Because this recipe and the next represent impartial mitosis, they are inherently unstable, uneasy compromises between prose form and complete mitosis. The result is that their style is inconsistent. For example, the choice of the ingredients to be privileged by surrounding white space is strictly arbitrary: from a structural point of view there is no reason why the wheat germ and paprika should not be so handled.

Note that unary operations applied to terminal ingredients are given prefix treatment: "1 chopped bell pepper" rather than "1 bell pepper, chopped".

Sample 6: Health Food Imperative Style II.



Clam Sauce with Garlic and Wine

4 servings
average serving = approx. 12 g usable protein
28-34% of daily protein allowance

Start cooking:
1/2 lb spaghetti

Drain juice from 2-3 8-oz cans minced clams and set aside.

Sauté:
1/4 cup olive oil
2 cloves garlic, minced

Stir in:
clam juice
3/4 cup chopped parsley
2 tbsp white wine
1 tsp basil
1/2 tsp salt
dash pepper

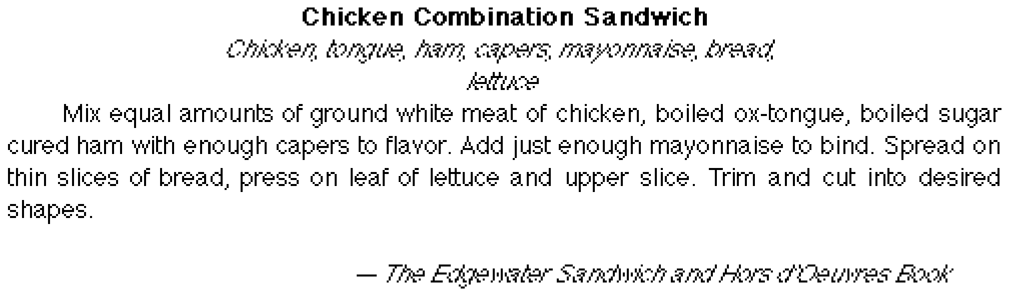
Now add clams and heat through while you drain the spaghetti. Serve over spaghetti. A special dish that is no trouble at all! For a feast, include garlic bread and Caesar salad.

— *Diet for a Small Planet*

An interesting comparison with the preceding recipe. Here the shopping list is engaged in a life-and-death struggle with the recipe itself. Culinary operations have been reduced to mere italicized comments on the ingredient list.

As in the previous example, incomplete mitosis leads to inconsistency. After having established the possibly useful rule that operators are italicized while operands are not, it throws the distinction away and breaks down into ordinary prose at the end. Even before that, the rule was applied inconsistently: note the italicized can of clams. The alternation between chitchat ("no trouble at all!") and recipe ("garlic bread and salad"), and the vacillation between postfix and prefix treatment of unary operations on terminals ("garlic, minced" vs. "chopped parsley") are additional confusions. All of this makes for very slow cooking.

Sample 7: Semi-professional Style.



Chicken Combination Sandwich
Chicken, tongue, ham, capers, mayonnaise, bread, lettuce

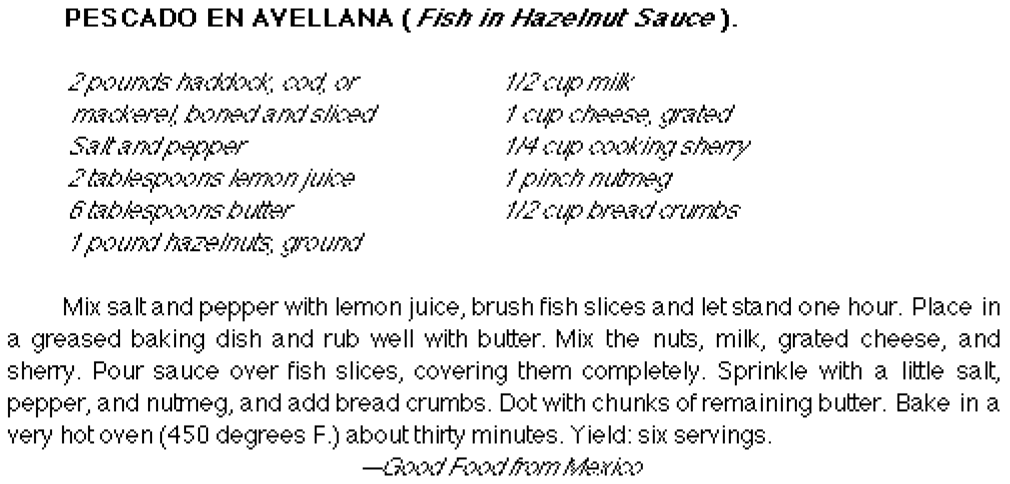
Mix equal amounts of ground white meat of chicken, boiled ox-tongue, boiled sugar cured ham with enough capers to flavor. Add just enough mayonnaise to bind. Spread on thin slices of bread, press on leaf of lettuce and upper slice. Trim and cut into desired shapes.

— *The Edgewater Sandwich and Hors d'Oeuvres Book*

Here we have another early form of mitosis. Instead of bold face, we have italics. Instead of marking the ingredient list in the recipe itself, it has been duplicated and placed at the head of the recipe.

As in the preceding examples, this form of mitosis is relatively innocuous. Nothing is left out of the body of the recipe; the ingredient list is totally redundant. Because this is a recipe for professional cooks, formal measurements have been dispensed with.

Sample 8: American Standard Style.



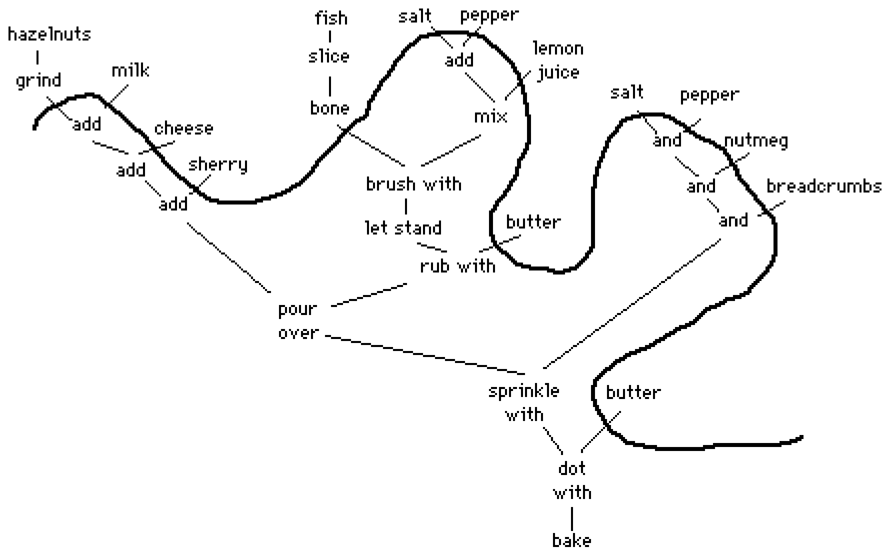
PESCADO EN AYELLANA (*Fish in Hazelnut Sauce*).

2 pounds haddock, cod, or mackerel, boned and sliced
Salt and pepper
2 tablespoons lemon juice
6 tablespoons butter
1 pound hazelnuts, ground
1/2 cup milk
1 cup cheese, grated
1/4 cup cooking sherry
1 pinch nutmeg
1/2 cup bread crumbs

Mix salt and pepper with lemon juice, brush fish slices and let stand one hour. Place in a greased baking dish and rub well with butter. Mix the nuts, milk, grated cheese, and sherry. Pour sauce over fish slices, covering them completely. Sprinkle with a little salt, pepper, and nutmeg, and add bread crumbs. Dot with chunks of remaining butter. Bake in a very hot oven (450 degrees F.) about thirty minutes. Yield: six servings.

—*Good Food from Mexico*

We come now to the quintessential contemporary recipe. Mitosis has reached completion; the deep structure of the recipe has been torn asunder, the leaves and twigs of the tree torn off and placed in a preamble to the rest:



I find nothing sadder than this total disruption of the recipe's integrity. Imagine how far mathematics would have progressed with a notation which wrote "a+b.(-c)+d" as:

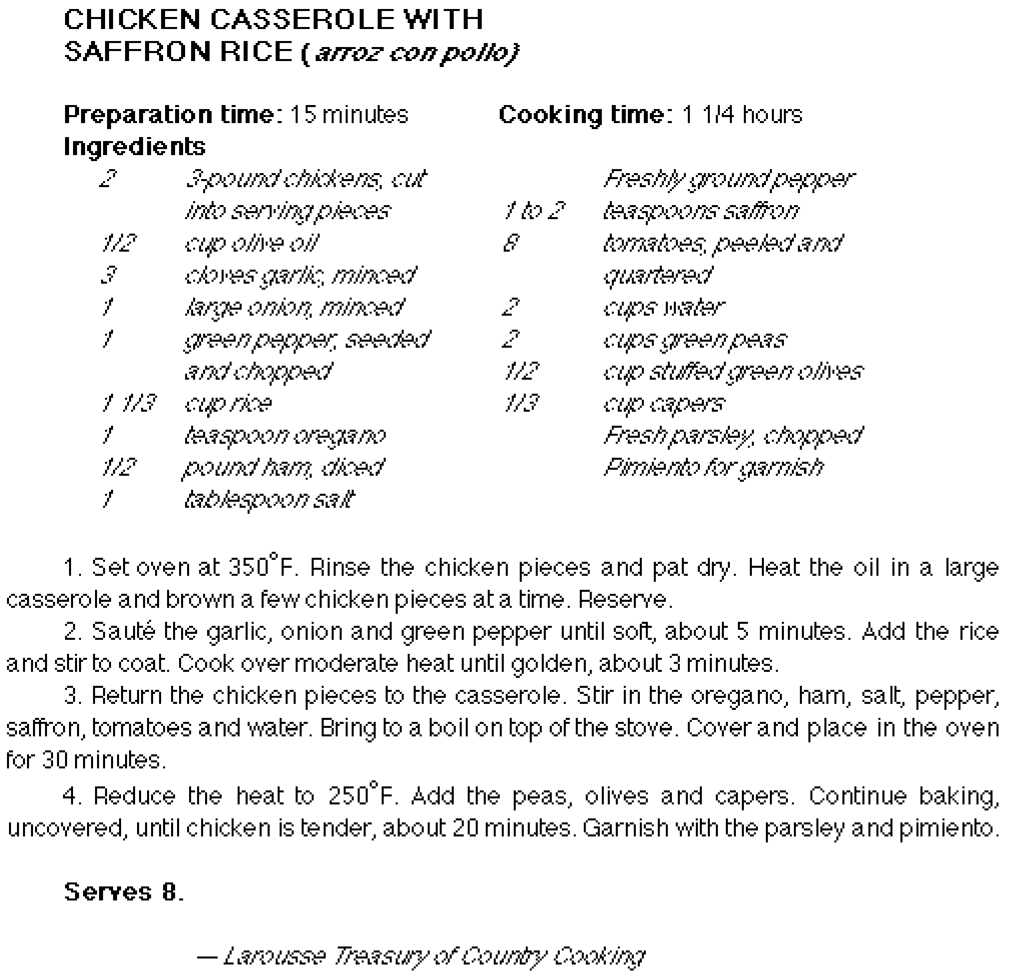
```
OPERANDS
          a
          b
          c, -
          d

          OPERATIONS
          +, ., -, +
```

Yet this is exactly what mitotic recipes do. The eye of the harried cook must constantly shift back and forth between the body of the recipe and the ingredient list. "Brush the fish. What fish??? Oh, the 2 pounds of fish that I have boned and sliced. OK, fish brushed. Now what? Add the salt to the pepper —no, I did that already."

This ping-pong effect was not a property of the recipes we have examined earlier, despite their mitotic tendencies. In them, the scission was performed by an invasion of white space (as in sample 5) rather than by a re-ordering, or else the material split off was strictly redundant, a sort of index to the recipe (as in sample 7).

It is obviously convenient to have an ingredient list, but it is not worth dismembering the recipe to get it. One of the nice features of RxOL is that the ingredient list falls out naturally as a by-product; ingredients line up effortlessly on the left.

Sample 9: The Stepwise Variation.



CHICKEN CASSEROLE WITH SAFFRON RICE (*arroz con pollo*)

Preparation time: 15 minutes **Cooking time:** 1 1/4 hours

Ingredients

2	*3-pound chickens, cut into serving pieces*
1/2	*cup olive oil*
3	*cloves garlic, minced*
1	*large onion, minced*
1	*green pepper, seeded and chopped*
1 1/3	*cup rice*
1	*teaspoon oregano*
1/2	*pound ham, diced*
1	*tablespoon salt*
	Freshly ground pepper
1 to 2	*teaspoons saffron*
8	*tomatoes, peeled and quartered*
2	*cups water*
2	*cups green peas*
1/2	*cup stuffed green olives*
1/3	*cup capers*
	Fresh parsley, chopped
	Pimiento for garnish

1. Set oven at 350°F. Rinse the chicken pieces and pat dry. Heat the oil in a large casserole and brown a few chicken pieces at a time. Reserve.

2. Sauté the garlic, onion and green pepper until soft, about 5 minutes. Add the rice and stir to coat. Cook over moderate heat until golden, about 3 minutes.

3. Return the chicken pieces to the casserole. Stir in the oregano, ham, salt, pepper, saffron, tomatoes and water. Bring to a boil on top of the stove. Cover and place in the oven for 30 minutes.

4. Reduce the heat to 250°F. Add the peas, olives and capers. Continue baking, uncovered, until chicken is tender, about 20 minutes. Garnish with the parsley and pimiento.

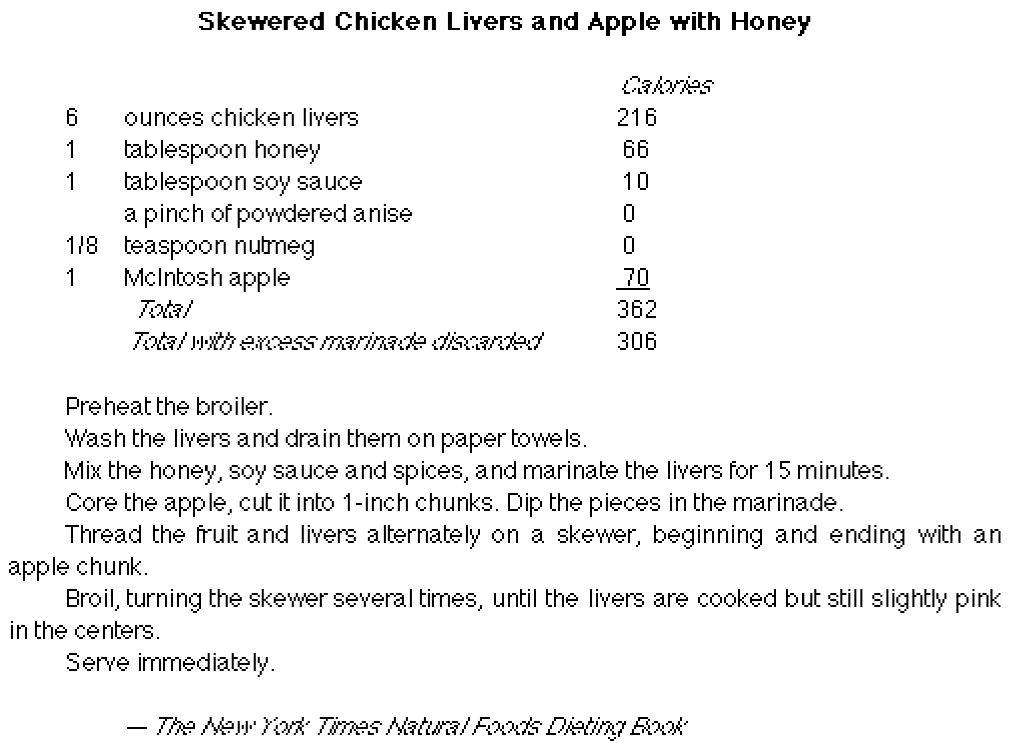
Serves 8.

— *Larousse Treasury of Country Cooking*

This is a variation on the Standard Style, included mainly to invite consideration of what constitutes a "step" in a recipe. The recipe at hand is typical in that its breakdown into steps is strictly arbitrary. One could just have well have added an extra step starting when the rice is added, or when the rice is brought to a boil, or at any one of six other places. Conversely, one might just as well have reduced the number of steps to three, or two, or one. Without an analysis of the recipe's deep structure, step delimiting is a futile, arbitrary business.

Once the tree structure is taken into account, however, two very natural definitions of a step emerge. A "small" step is simply an operator: slicing, simmering, adding, all the primitive culinary operations. This definition is useful in automatically calculating the complexity rating of a recipe. A "large" step is defined technically as a compound right operand: on the tree structure, it is any nonterminal which is the right operand of a binary operator. It corresponds to a place where the cook must put aside the operand he is working with to prepare a whole new subrecipe which is only later merged with the main line. An example from the Fish with Hazelnut Sauce recipe is the fish itself: after preparing the sauce, one sets it aside to prepare the fish, then pours the sauce over the fish. It is these compound right operands which are optionally marked with braces in RxOL, and which determine the breakdown into steps with FNL.

Sample 10: Diet Style I.



Skewered Chicken Livers and Apple with Honey

		Calories
6	ounces chicken livers	216
1	tablespoon honey	66
1	tablespoon soy sauce	10
	a pinch of powdered anise	0
1/8	teaspoon nutmeg	0
1	McIntosh apple	70
	Total	362
	Total with excess marinade discarded	306

Preheat the broiler.

Wash the livers and drain them on paper towels.

Mix the honey, soy sauce and spices, and marinate the livers for 15 minutes.

Core the apple, cut it into 1-inch chunks. Dip the pieces in the marinade.

Thread the fruit and livers alternately on a skewer, beginning and ending with an apple chunk.

Broil, turning the skewer several times, until the livers are cooked but still slightly pink in the centers.

Serve immediately.

— *The New York Times Natural Foods Dieting Book*

The breakdown into steps here is less arbitrary than usual, but is still inexact, and their representation as unindented paragraph breaks is unsightly. The recipe is included for the way it exploits the shopping list to provide some nutritional information. Because RxOL is a formal language, nutritional information can be automatically generated at will, and need not clutter up the recipe itself.

Sample 11: Szechwan Verbose.

GRAND DUKE'S CHICKEN WITH PEANUTS (gongbao jiding)

There is some debate as to just which grand duke gave his title to this celebrated chicken dish, usually called simply Grand Duke's Chicken on Chinese-American menus. As in Europe, it was not uncommon for a celebrated Chinese gourmet to allow his name to be used for a classic dish, one his sensitive criticism may have helped perfect. Whoever he was, our anonymous grand duke must have been a gentleman of impeccable taste, for his chicken with peanuts is one of the great dishes of Szechwan. Many gourmets feel it epitomizes the true taste of the province.

Debate also surrounds the composition of the dish; some cooks use only dried red peppers, while others, including Mrs. Chiang, add green peppers. But everybody uses peanuts.

When she can get them, Mrs. Chiang likes to use hot green peppers instead of the regular round variety. Together with the dried red ones, they produce a very hot chicken dish. Her *gongbao jiding* is light and aromatic, as brilliantly flavored as it is lovely to look at.

The peanuts for this dish must be fresh; neither roasted nor salted ones will do. You can get fresh peanuts at health food stores as well as at Chinese markets.

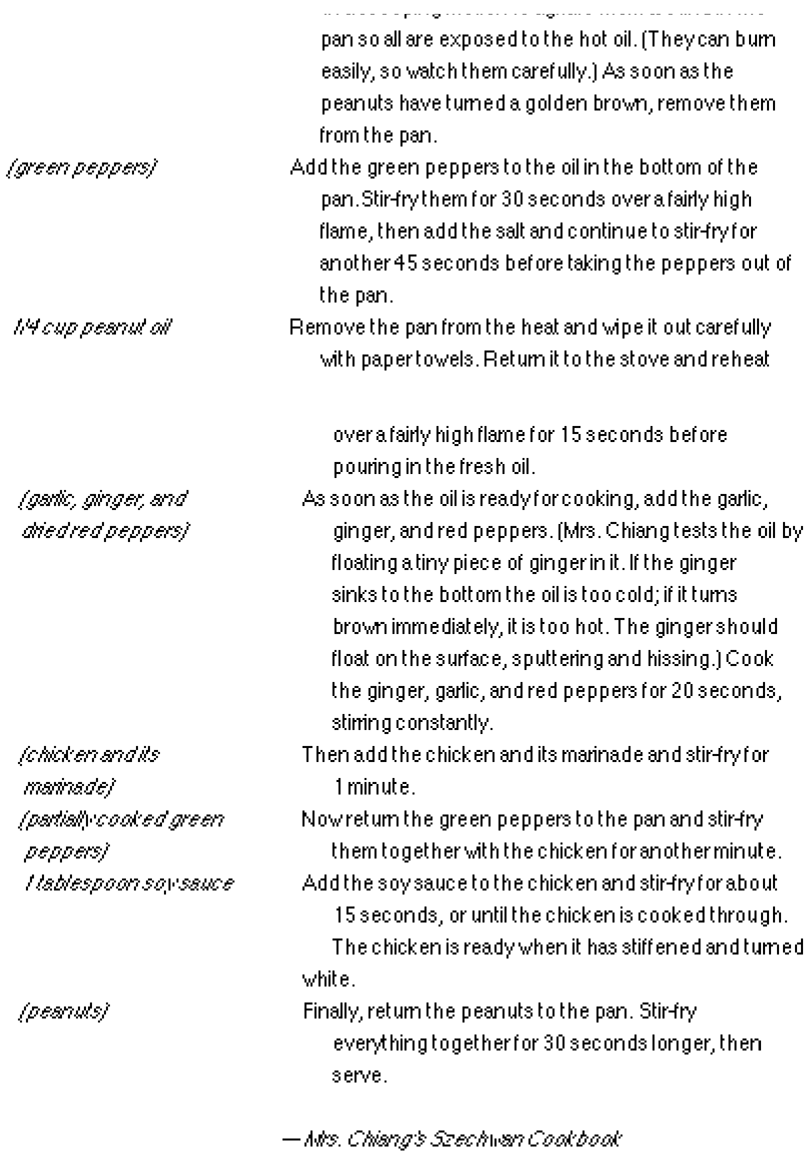
Perfectly prepared, a *gongbao jiding* will have very little sauce, although it may be covered with a thin film of oil. Szechwanese food occasionally seems oily; this is not a flaw but a sign that the cook respected his ingredients enough not to adulterate them with any prepared sauce.

PREPARATION

1/2 cup fresh *peanuts* *1 cup boiling water (optional)*	If the peanuts still have their dark red skins on, put them in a small bowl and pour boiling water over them. Let the peanuts soak for about 3 minutes, then drain them; the skins will practically pop off. (If the peanuts have already been skinned, omit this step.)
1 large whole chicken breast (about 1 pound)	Remove all the skin and bones from the chicken breast and cut the meat into cubes roughly 1 inch in diameter.
1 1/2 tablespoons soy sauce *1/2 teaspoon granulated sugar* *1 teaspoon sesame oil* *1 teaspoon Chinese rice wine or cooking sherry* *1 egg white* *1 scant tablespoon cornstarch*	Put the chicken pieces in a bowl and add the soy sauce, sugar, sesame oil, wine, egg white, and cornstarch. Mix thoroughly and set it aside to marinate while you prepare the other ingredients.
2 green peppers or § 3 large hot green peppers	Wash the peppers and cut them into squares that are approximately the same size as the chicken pieces.
10 cloves garlic	Smash the garlic cloves with the flat of your cleaver, then peel. Chop the garlic into little pieces, about the size of a match head.
1/2-inch piece fresh ginger	Peel the ginger and mince it into slightly smaller pieces than the garlic.
§ 5 dried red peppers	Cut each of the red peppers into about 4 pieces.
2 scallions	Clean the scallions, then chop them, both green part and the white, crosswise into 1/4-inch pieces. Add the scallion to the chicken.

COOKING

3 tablespoons peanut oil	Heat you wok or pan for 15 seconds over a medium flame before you pour in the oil. The oil should be hot enough to cook with when the first tiny bubbles form and a few small wisps of smoke appear.
(peanuts)	When the oil is ready, add the peanuts. Stir-fry them for 2 or 3 minutes, using your cooking shovel or spoon in a scooping motion to agitate them around in the



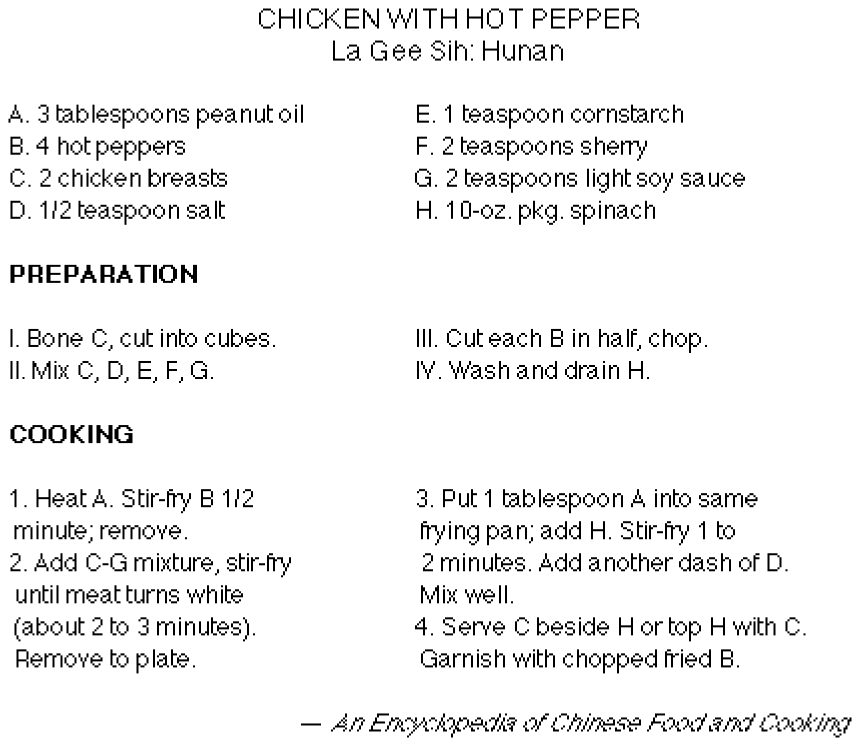
	pan so all are exposed to the hot oil. (They can burn easily, so watch them carefully.) As soon as the peanuts have turned a golden brown, remove them from the pan.
(green peppers)	Add the green peppers to the oil in the bottom of the pan. Stir-fry them for 30 seconds over a fairly high flame, then add the salt and continue to stir-fry for another 45 seconds before taking the peppers out of the pan.
1/4 cup peanut oil	Remove the pan from the heat and wipe it out carefully with paper towels. Return it to the stove and reheat over a fairly high flame for 15 seconds before pouring in the fresh oil.
(garlic, ginger, and dried red peppers)	As soon as the oil is ready for cooking, add the garlic, ginger, and red peppers. (Mrs. Chiang tests the oil by floating a tiny piece of ginger in it. If the ginger sinks to the bottom the oil is too cold; if it turns brown immediately, it is too hot. The ginger should float on the surface, sputtering and hissing.) Cook the ginger, garlic, and red peppers for 20 seconds, stirring constantly.
(chicken and its marinade)	Then add the chicken and its marinade and stir-fry for 1 minute.
(partially-cooked green peppers)	Now return the green peppers to the pan and stir-fry them together with the chicken for another minute.
1 tablespoon soy sauce	Add the soy sauce to the chicken and stir-fry for about 15 seconds, or until the chicken is cooked through. The chicken is ready when it has stiffened and turned white.
(peanuts)	Finally, return the peanuts to the pan. Stir-fry everything together for 30 seconds longer, then serve.

—Mrs. Chiang's Szechwan Cookbook

It is hard to imagine a greater contrast to the concise professionalism of the Chasseur recipe from the Répertoire de la cuisine than this bewilderingly verbose tract. The Chasseur dish was of about the same complexity as this one, yet its recipe contains about as many lines as this one does pages! Instead of treating the reader as an experienced professional, this text treats him as a bungling idiot. The actual preparation of the dish is camouflaged behind a smoke-screen of anecdotes, sociology, and minute details of culinary technique.

It is noteworthy for its attempt to mitigate the harmful effects of mitosis: instead of placing the ingredient list at the head of the recipe, it is distributed along the margin, with each ingredient placed close to where it is referenced in the text. (The extreme length of the recipe make such an expedient almost a necessity.) Note also the use of pseudo-operands: when an ingredient has already been referenced once, future appearances in the ingredient list are placed in parentheses. This is very similar to the colon notation in RxOL.

Sample 12: Oriental Predicate Calculus



CHICKEN WITH HOT PEPPER
La Gee Sih: Hunan

A. 3 tablespoons peanut oil
B. 4 hot peppers
C. 2 chicken breasts
D. 1/2 teaspoon salt
E. 1 teaspoon cornstarch
F. 2 teaspoons sherry
G. 2 teaspoons light soy sauce
H. 10-oz. pkg. spinach

PREPARATION

I. Bone C, cut into cubes.
II. Mix C, D, E, F, G.
III. Cut each B in half, chop.
IV. Wash and drain H.

COOKING

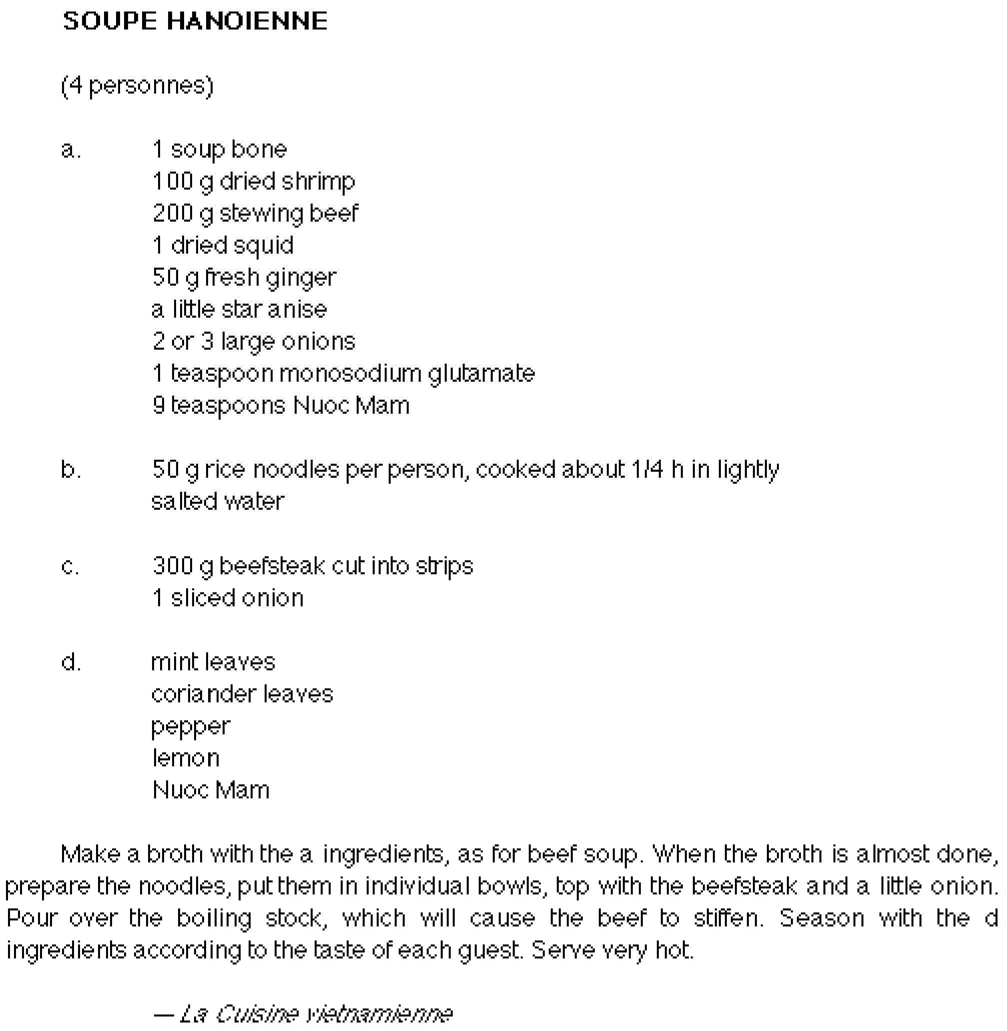
1. Heat A. Stir-fry B 1/2 minute; remove.
2. Add C-G mixture, stir-fry until meat turns white (about 2 to 3 minutes). Remove to plate.
3. Put 1 tablespoon A into same frying pan; add H. Stir-fry 1 to 2 minutes. Add another dash of D. Mix well.
4. Serve C beside H or top H with C. Garnish with chopped fried B.

— *An Encyclopedia of Chinese Food and Cooking*

This recipe style carries mitosis to its logical conclusion. Not only must you refer back to the shopping list for the quantity of an ingredient, but also for its name. As in many oriental styles, the unary operators on the leaves (along with certain binary operators at the top of the tree regarded as "to be done in advance") are placed in a section by themselves under the rubric "preparation". This is chaos, but is no worse than the American Standard practice of attaching them to the shopping list. As with the stepwise variation examined earlier, the seemingly carefully delimited "steps" are devoid of any rational criterion for their existence.

This recipe and the next one have been included in our survey to illustrate the oriental technique of using ingredient variables: the compound operands from the preparation phase are given one-character identifiers, which are then used to reference them during the actual cooking. Presumably this is an implicit acknowledgement that having two copies of each ingredient name around slows down the cooking process—a slow-down that is unacceptable during fast-paced stir-frying.

Sample 13: Indochinese Algebraic.



SOUPE HANOIENNE

(4 personnes)

a. 1 soup bone
100 g dried shrimp
200 g stewing beef
1 dried squid
50 g fresh ginger
a little star anise
2 or 3 large onions
1 teaspoon monosodium glutamate
9 teaspoons Nuoc Mam

b. 50 g rice noodles per person, cooked about 1/4 h in lightly salted water

c. 300 g beefsteak cut into strips
1 sliced onion

d. mint leaves
coriander leaves
pepper
lemon
Nuoc Mam

Make a broth with the a ingredients, as for beef soup. When the broth is almost done, prepare the noodles, put them in individual bowls, top with the beefsteak and a little onion. Pour over the boiling stock, which will cause the beef to stiffen. Season with the d ingredients according to the taste of each guest. Serve very hot.

— *La Cuisine vietnamienne*

Another example of the oriental practice of using ingredient variables. The method is not applied consistently here: when the ingredient name is short, it is simply repeated, as happens to b and c.

Sample 14: RxOL Postfix Style I.

```
 *hazelnuts =grind
 *milk /add
 *cheese /add
 *sherry /add
{ *fish
{ *salt
 *pepper /add
 *lemon juice /mix with} /brush with =let stand
 *butter /rub with} /pour over
{ *salt
 *pepper /and
 *nutmeg /and
 *bread crumbs /and} /sprinkle with
 *butter /dot with =bake
```

As mentioned previously, RxOL, the formal language devised at Culinary Software Systems, was designed to reflect as closely as possible the deep structure of recipes. Its grammar is the grammar of cooking itself. It uses postfix, or "Reverse Polish," notation to describe recipes exactly as logicians use it to describe inferences or computer scientists use it to describe computations.

Before presenting a complete RxOL recipe, we consider the RxOL expression for the tree which was drawn earlier. As can be seen, this expression bears an exact, one-to-one relationship to the tree it represents. In fact, it is generated by traversing the tree and writing down the name of each node after traversing its operands. This postfix form corresponds perfectly to the natural rhythm of cooking: first you take one ingredient, then you take another ingredient, then you put them together. First you take an onion, then you take some oil, then you sauté. First you take some sugar, then you take some butter, then you cream then together.

The curly braces in the formula require some explanation. Unlike infix notation, postfix notation does not require parentheses to distinguish different interpretations of a formula. The infix form a+b+c can mean either a + (b+c) or (a+b) + c; the corresponding postfix formulas, however, are unambiguous even without parentheses: abc++ and ab+c+. Nevertheless, the human eye has some difficulty grouping postfix forms when the second operand of a binary operator is compound, so RxOL provides the option of bracketing such operands. The bracketed form of abc++ would be a{bc+}+.

Sample 15: RxOL Postfix Style II.

```
<* 1="" 10="" 15="" 30="" 50="" 100="" 125="" 150="" chicken="" chasseur=""
(poulet="" sauté="" chasseur).="" *chicken,="" kg="cut" up="" {="" *oil=""
*butter="" and="" }="" in,="" till="" three-quarters="" cooked=""
*mushrooms,="" g="slice" add="cook" *:chicken="remove" *white="" wine,=""
ml="" *shallot,="" *thickened="" rich="" veal="" gravy,="" *tomato=""
sauce,="" s="" *brandy,="" *parsley,="" *chervil,="" *tarragon,="" pour=""
over="" *parsley="mince" sprinkle="" with="" class="jop-noMdConv">
```

This sample is a complete RxOL recipe which shows how measurements and annotations are incorporated, and how the recipe as a whole is structured. This style could be thought of as post-mitotic, as the re-unification of the recipe under the aegis of the ingredient list. The ingredients are still lined up in a nice column on the left, but the operations on them are immediately attached, hanging off to the right. As we shall argue when considering recipological axiology, this permits a rapid scanning of the recipe and greatly facilitates cooking.

Sample 16: Cocina Natural Language Style.

```
Chicken with Rice (Pollo con Arroz).

Ingredients:
        1       chicken.
        1 clove garlic.
        0.5     onion.
        4       peppercorns.
                salt.
                cumin.
                fat.
        250 g   rice. Soak, 15 min.
        2       tomatoes. Chop.
        1 L     water.
        2       sweet peppers, green. Slice.
        1       chicken liver, from chicken. Grind.

Preparation:

[A]     Add onion, peppercorn, salt, and cumin to garlic. Grind.
[B]     Spread chicken with [A]. Brown in fat, lightly. Add rice. Brown,
        lightly. Add tomatoes. Sauté, 5 min. Add water and sweet
        peppers. Cover. Simmer, till done. Add chicken liver.
```

Despite its many advantages, RxOL does impose a certain learning burden on the user. To ease the transition, Cocina provides an alternative representation called formalized natural language (FNL). The translation from RxOL to FNL is fully automated; the Cocina user can toggle back and forth between them instantaneously.

Although it suffers from some of the drawbacks of mitosis, this style benefits from the formal underpinnings of RxOL itself, ensuring a consistency which makes it relatively easy to cook from. Steps are always delimited sensibly, and the structure of the recipe is fairly apparent despite the prose-like form. The mitotic familiarity is an asset, and does allow the new user to start using Cocina at once.

Recipological Axiology

The overriding criterion by which a recipe style must be judged at this point in history is surely the speed with which a dish can be prepared using a recipe in that style, for the restricted time available for cooking is the limiting factor in the culinary experimentation of the average home. The desirable qualities which a speed-oriented recipe style should possess may be summarized as the four C's:

Recipes should be consistent. As in all other technologies, consistency is the key to a good user interface. When the cook's expectations are constantly being shattered by random changes in nomenclature, organization, typography, measurement systems, and so forth, the process of cooking is impeded.

Recipes should be concise. Only if the eye can take in the entire recipe in a few quick glances can maximum efficiency be obtained. Fast look-ahead permits optimal concurrency in the processing—the second ingredient can be chopped while the first is being sautéed, and so on.

Recipes should be correct. Nothing is more frustrating than to get three-quarters of the way through a dish and discover that the author never tells you what to do with the breadcrumbs, or that the proportions he gave were manifestly erroneous. The authenticity of the recipe, or the correlation between the name and the dish, is another form of correctness.

Finally, recipes should be comprehensive. This means not only that collections of recipes should be thorough in their coverage of a given culinary domain, particularly with respect to variants, but also that the style should be flexible enough to embrace any culinary style and to include any information the recipe author feels he must provide (no matter how ill-advised its inclusion).

RxOL has been designed with those four qualities in mind. Because RxOL is based on a theory of what a recipe is, it takes a universal approach to notation. Because it is a formal language susceptible of automated analysis, many of its features, such as vocabulary and measurements, can be guaranteed consistent. Inconsistencies such as using "chop finely" instead of "mince" can be eliminated automatically.

The RxOL form fosters concision because it encourages authors to write down just those operands and operators which are involved in the dish, neither more nor less. RxOL syntax checkers automatically guarantee the syntactic correctness of the recipe, that is, that the sequence of operands and operators given fits together properly. In the future, semantic checkers will be able to check for suspected semantic inconsistencies, such as pouring ice cream over a steak recipe, as well.

Finally, RxOL provides a notation for variants and comments, to encourage comprehensive coverage and to permit the inclusion of any type of miscellaneous material without disrupting the structure of the recipe. Future generations of the Cocina software will further facilitate the division of background information from recipe by providing CD-Rom technology to allow instant access to immense encyclopedic information relevant to the recipe. Clicking on "lamb," for instance, will instantaneously bring up a detailed article on lamb in cooking, complete with a chart of cuts, while clicking on "sauté" will display a detailed article on sautéing.

Chapter 2. The Goals of Culinary Software

Is cooking fun?

Those who have savored the joy of sitting down to a meal they have prepared, whose adrenaline has surged at the challenge of mastering a complex menu in time for a deadline, whose soul has relaxed at the gentle rhythms of kneading bread or chopping onions, will hesitate before answering "No."

And yet the great masses of cookbooks for people who hate to cook, of 10-minute gourmet cookbooks, of Chinese (or French or Indian or Mexican) cuisine made easy cookbooks, the vast sums of money spent on dysfunctional appliances and gadgets for the kitchen, the disastrous rise of prepared and fast foods, all argue that for many people the answer is indeed "No."

I take as a given that eating is fun, that culinary exploration is fun, that thinking about recipes and inventing new ones is fun. I leave open the question of whether or not the manual labor underlying such enterprises is fun. Let readers consider carefully whether, given the choice tomorrow evening, they would spend 30 minutes preparing a Poulet sauté au basilic by hand, or would instead spend 30 seconds punching in their order to their robot. Cooking may well be a recreation, like tennis or crossword puzzles, where automation would miss the point; on the other hand, it may well fall into the category of drudge work best left to the droids.

So I do not say that the goal of computerized cooking is to free cooks from cooking, but rather to raise the level at which cooking is conceived, to make it possible to think about recipes in more abstract terms than just how a given recipe can be put together.

Programming environments for the kitchen.

Because RxOL is a formal language, the vast array of computer techniques which have been developed for such formal languages can be applied to it directly. Eventually, advances in natural language processing may neutralize this advantage, so that the only advantages of RxOL would be those for human readers, as discussed in the preceding section. For the next few years, however, RxOL will enjoy a clear advantage.

The culinary tools opened up by RxOL are limited only by the imagination. In this section I have chosen fifteen representative tools to illustrate what is possible. None of these tools pushes the state of the art in computing; all of them have well-known analogues in programming languages and data base management. Most of them fall into the traditional classification of software tools as editors, compilers, static program analyzers, and dynamic performance monitors. Eight of these tools are currently implemented in the Cocina system.

> **Editor.** The Cocina editor makes use of the latest techniques developed for manipulating programming languages and other formal languages. Freeing the author from the necessity of thinking of his recipe as a text, that is, as an unstructured series of characters, it instead supports and encourages viewing the recipe as a tree. The editor, in fact, uses the RxOL grammar to ensure that it is impossible to enter a syntactically incorrect recipe, thus freeing the author from the tedium of doing his own proofreading.

> The most important component of the editor is the parser, whose job it is to convert the input typed by the recipe author into a tree representation which can be used by the other tools in the Cocina environment. The editor also incorporates a number of other simple but effective subtools, such as a scaling tool which automatically adjusts a recipe to produce any given amount, and a measurement conversion tool which will, when absolutely required, convert metric units into those other ones.

> **Pretty printer.** Like the parser, the pretty printer is normally considered a component of the editor, but we single it out here to emphasize that once a recipe is available in RxOL form, it can be displayed in any of a number of different ways. RxOL Postfix and FNL should really be

> thought of as just two pretty-printing options. A third option is to display the RxOL tree graphically in two dimensions. This offers several advantages, chief among them being the extreme efficiency with which the human eye can process graphical structures. Its primary disadvantage is the small amount of data which it allows displaying on a reasonably-sized screen.

> **Semantic analyzer.** The Cocina semantic analyzer plays an important supporting rôle for many of the following tools. It is functionally similar to the type-matching component of a compiler, but is best understand in terms of the analysis of meaning as understood in computational linguistics, which distinguishes among syntactically incorrect sentences ("buttery an the poked") from syntactically correct, semantically incorrect sentences ("colorless green ideas sleep furiously") from sentences that are both semantically and syntactically correct ("valiant red heroes sleep quietly"). In computational linguistics, the analysis of a sentence proceeds by examining the fundamental elements of meaning, or "sememes," attaching to each word in the sentence, and then searching for sememic conflicts. Thus in our second example, the sememe inanimate attaching to "ideas" conflicts with the sememe animate attached to "sleep." In recipes, correct semantics is a matter of combining ingredients and operations which harmonize with each other. In the hypothetical recipe " *ice cream *butter /sauté in," the sememe "destroyable by heat" in "ice cream" conflicts with the sememe "applies to things resistant to heat" in "sauté."

> In recipology we may make a simplifying assumption which is unavailable to the natural-language semiologist, namely, that all meaningful combinations of terms occur someplace in the data base. The natural-language semiologist cannot infer from the fact that he has no text which refers to brushing butterflies that the concept contains a semantic conflict. The culinary semiologist, however, because of the relatively vast quantity of data he has at his disposal relative to the small number of terms involved, is justified in concluding that if there are no examples of frying ice-cream in his data base, it is probably a "meaningless" combination. Thus all queries about the "meaning" of a recipe can be reduced to statistical questions about the existent data base: "Have sage and cherries ever occurred together in a recipe? Have potatoes ever been used in a stir-fry?" This approach has its limits, of course; at some point someone must discover delicious combinations of ingredients that were previously considered meaningless.

> **New recipe generator.** Once an adequate semantic analyzer is available, it will be possible to fill in the gaps in the world's cuisine by detecting holes in the semantic fabric of its recipes. The NRG might observe, for example, that tarragon correlates well with pork, and pork with mustard, and mustard with tarragon, but that there is no recipe that combines all three, and so generate a recipe for Côtes de porc sauté à l'estragon au diable. By decreasing the value of the accepted correlation coefficient threshhold, recipes of any desired level of audacity could be generated. Unfortunately, given the methodology employed, there is no way of guiding the NRG in creating radically new dishes. If the correlation coefficient threshhold is set to zero, any combination of ingredients will be accepted, no matter how awful.

> **Nutritional analyzer.** This and the next two tools fall into the category which computer science calls "program analysis tools." Given a program text (or recipe), they derive a metric describing some aspect of the text. The output of all these tools will be available for other tools, such as the Menu Suggester and the Data Base Manager.

> The nutritional analyzer's task is to generate a summary of the nutrients provided by a given recipe. Of particular interest will be the kJ counter tabulating the food energy value of the dish, extremely useful for calculating one's food energy intake.

> **Cost analyzer.** Closely related to the nutritional analyzer is the cost analyzer, which will use the price and amounts of a recipe's ingredients to calculate its cost, including its cost per gram. Unlike the nutritional information, the cost information is somewhat time- and space-dependent, so will require occasional updates to remain accurate.

> **Complexity analyzer.** In the past, the evaluation of a recipe's complexity and of the time required for its execution has been a matter of expressionistic intuition on the part of the recipe's author. Where one author might assess the time required for Poulet sauté au basilic as 20 minutes, another might list it as 75 minutes. Where one might rank Boeuf bourgignon as a finger exercise for beginners, another might treat it as an intimidating enterprise only to be undertaken by professionals.

> Capturing the deep structure of a recipe in an RxOL formula, however, lays the groundwork for an impartial, scientific analysis of the complexity of that recipe. The time for the recipe as a whole is simply the sum of the times for each of its component operations. Account must be taken, however, of the increased complexity of "branching" recipes: given two recipes with the same number of operands and operators, which in and of themselves are of equal complexity, that recipe which has the greater number of compound right operands will be the more challenging: it is always easier to keeping adding things to one container than to run off and find another container and begin putting together a new compound operand in it.

> **Shopping list generator.** Simple in its conception, and in its execution as well once a RxOL formula is available, the shopping list generator is nonetheless one of the most radical of the Cocina tools as far as its influence on the day to day operations of the practicing chef goes. Our study of The Great Mitosis has already shown the importance commonly attached to the ease of generating shopping lists: a whole generation of cookbook authors has willingly mutilated their recipes to facilitate such generation. No one who has pointed the mouse at a week's worth of recipes and then had Cocina generate a shopping list sorted by grocery store section would happily go back to copying the list down by hand.

> For lack of a better place to mention it, we mention here that the concept of disposable recipes is another profound liberation for the practicing chef. In the past, one had the choice of choosing a week's recipes from one or two sources, or of carrying an armload of books around, or of copying the selected recipes by hand or by means of xerography. Once the entirety of the world's cuisine is on-line and attached to a printer, however, it becomes a trivial

> matter to generate a customized pamphlet of recipes for the week. No more fear of spattered pages, no more cumbersome plexiglass book stands: toss the recipe on the counter, coat it with tomato sauce and chocolate, then toss it in the garbage can.

> **Menu suggester.** Like disposable recipes, the Menu Suggester is a sleeper among culinary software. Surprisingly, it turns out that selecting menus is as onerous or more so as preparing them. Given a larder and a recipe, most cooks are quite willing to put the meal together; the hard part is browsing through the cookbooks searching for appropriate recipes.

> In one simple implementation, the Menu Suggester is just a random recipe generator: it picks some recipe at random, and proposes that it be added to the week's list. In future instantiations, it will be a simple matter to add more intelligence to the Menu Suggester, and to use the recipe analysis tools to balance the recipes nutritionally, or keep them within a budget, or keep their complexity metric under a certain threshhold, or see that they provide a balanced input stream—the leftovers from one recipe providing the "cooked chicken" needed by the next, and so on.

> **Data base manager.** The idea of a culinary data base was an early inspiration for Culinary Software Systems. The idea was to free cooks of the necessity of choosing recipes and generating a shopping list before going to the supermarket by allowing them to purchase ingredients on impulse and then query the data base to see what could be done with the assorted ingredients they had brought home. Having purchased a chicken, some fish, some tomatoes, some red peppers, and some noodles, the data base might suggest "Poulet sauté à la Bohémienne" and "Soupe de poisson à la Marseillaise."

> Given the Menu Suggester and the Shopping List Generator, however, the burden of planning ahead does not seem as great as it once did. Still, it seems obvious that one should be able to interrogate the online recipes in almost any conceivable way. It would be wonderful to say "Give me all the recipes which call for pork and tomatoes with a complexity metric of less than 7 and a food energy density of less than 6 kJ/g."

> **Interactive encyclopedic browser.** In section 1 I argued that despite its interest, the great body of sociological, biological, and instructional materials which currently bog down recipes belongs elsewhere. This is that elsewhere. I shall not call it a "help facility," since I think that is an inherently pejorative term—a help facility is what you provide after you have botched your user interface. The tool in question here is more of a learning tool, a means of exploring the vast domains of culinary science. Integrated into the editor, this tool will permit the user, for example, to point to the word "sauté" in a recipe, and summon up a complete interactive tutorial on sautéing, taking advantage of the advances in computer interfaces and mass storage devices (CD-Roms) of the past few years.

> We are witnessing a radical re-thinking of what a text is. As has been pointed out before, users will not give up the familiarity of the printed text without substantial rewards; in the culinary realm the rewards are there. One can envision a future in which, for example, an

> Indonesion curry recipe appears in the middle of the screen, in all its RxOL purity. Along the right margin are a series of buttons, labeled "History. Sociology. Travelogue. Chitchat. Utensils. Technique. Photographs." Clicking on any of the buttons brings up further information on that aspect of the recipe. In this way, the recipe serves as the skeleton on which are hung all the other culinary viewpoints.

> But of course nothing prevents the user of such as system from turning it inside out—of placing, for example, the travelogue in the middle of the screen, and turning the recipes into one of the buttons on the side. Such is the protean nature of text in the hypertext era.

> **Robot instruction generator.** In the not too distant future it will doubtless be practical to build culinary machines capable of executing many of the necessary primitive operations needed for cooking—sautéing, measuring spices, adding liquids, controling temperatures, and so on. Given a RxOL recipe tree, generating the machine instructions to execute that recipe will be fairly straightforward. If we consider our analogy between RxOL and programming languages, we see that this tool corresponds exactly to the "code generation" phase of a traditional compiler. The major conceptual obstacle to implementing the RIG is the sequencing problem discussed in Chapter 3.

> The transformations which this combination of RxOL and some special-purpose hardware might effect on our concept of cooking are profound indeed. We can indeed envisage the day when one might simply type in the name of a recipe and have the robot cook it.

> **Interactive execution monitor.** Even in the absence of culinary robots, however, code generation can be useful. If the target machine language is English or another natural language, the recipe can be "executed" by a human being: the computer can, in real time, issue step by step commands such as "take a break for 10 minutes" or "chop an onion," effectively relieving the cook of the need to look at the recipe at all. The value of such a capability is greatly increased when several dishes are being prepared simultaneously—the computer can automatically merge the recipes involved and compute exactly what should be done when. Like all culinary code generators, the IEM depends on solving the sequencing problem.

> **Metarecipe generator or recipe unifier.** The problem of the taxonomy of dishes is a largely unaddressed one. Most cookbooks dealing with a genus of dishes single out some one species and treat the other members of the genus as "variants". This is of course totally capricious; there is no more reason to treat Pork with zucchini as a variant of Beef with zucchini than there is to treat Beef with zucchini as a variant on Pork with zucchini .

The original impulse for devising a formal recipe language was to provide a methodical framework for dealing with this problem. The idea was to find metarecipes, i.e. recipes containing variables, when the variable was bound to some particular term. The current RxOL approach uses ternary trees to list alternatives alongside each node.

Whatever the method used to represent variants, the function of the metarecipe generator is to factor out common subexpressions, that is, to identify duplicated subtrees and to merge the recipes containing them into a single metarecipe. This is isomorphic to the problem of classifying recipes according to the shape of their trees. (Note that the MG eliminates the operators add and and from the tree before performing such an analysis.) With such a tool, one would at last be able to make a scientific attempt at recipological taxonomy, to answer the question of how many kinds of recipes there are. This would be of tremendous benefit in learning recipes, since once a recipe type has been mastered, all recipes of the same shape come for free.

Chapter 3. An Introduction to RxOL

1. Operands and Operators

Ingredients are marked with bullets, as:

```
*onion, 500 g
*butter
```

Unary operations, i.e. those which are performed on a single (possibly compound) ingredient, are marked with equal signs, and are placed immediately after the ingredient to which they refer, as:

```
*onion =peel
```

Binary operations, i.e. those which combine two (possibly compound) ingredients to form a new, compound ingredient, are marked with slashes, and are placed immediately after the two ingredients which they unite, as:

```
*onion =peel
*butter /saute in
```

Ingredients may be marked as "pseudo-operands" by inserting a colon after the bullet; in this case they are ignored by the CSS shopping-list generator and other software. Operands may be marked as simultaneous by following their slash or equals-sign by a plus-sign; in this case they are to be performed at the same time as the preceding operator. Examples:

```
*chicken
*butter /saute in
*:chicken /remove (You don't want the software to think
you need another chicken!)
*cream /add =bring to boil =+beat (i.e. beat the cream
as you bring it to a boil)
```

2. Variants; Recipological Taxonomy

The topic of variants is one of the thorniest in recipe theory, involving as it does the problem of the taxonomy of recipes. In RxOL, variants are surrounded by brackets and separated by vertical bars, as:

```
 [ *onions |
 *scallions |
 *shallots ]
 *butter /saute in
```

This would be expressed in natural language as "saute onions or scallions or shallots in butter". Variants may be numbered using the symbol "#" and a list of numbers separated by commas. This is done in order to link variants with each other or with variant phrases in the recipe's name. When a single unary operator is enclosed in brackets, the meaning is that the operator is optional.

To see the relationship between variants and classification, consider the following two recipes:

```
 *pork *butter /sauté in *parsley /sprinkle with


 *steak *butter /sauté in *parsley /sprinkle with
```

In American Standard style, one of these two, on a purely arbitrary basis, would be chosen as the "main" recipe, and the other reduced to the status of a "variant":

```
Sauté the pork in butter. Sprinkle with parsley.
Variants:
Substitute steak for the pork.
```

In the formal language which preceded RxOL, and which was inspired by the predicate calculus, the variable term was given a name, and the list of values for the variable was provided separately:

```
 *a *butter /sauté in *parsley /sprinkle with


a =
         *pork
         *steak
```

Since the variable name is never referenced except in the definition of its values, this "closed" form was ultimately rejected in favor of an "open" form where the alternative values are listed in place:

```
[ *pork | *steak] *butter /sauté in *parsley /sprinkle with
```

If ingredients were the only RxOL items that could have variants, it would perhaps be clearer to write this as:

```
 *pork *steak /or *butter /sauté in *parsley /sprinkle with
```

However, unary and binary operators can have variants too:

```
 *pork [=julienne | =dice] *butter /sauté in
```

The introduction of "logical" binary operands such as "/or" does nothing to solve this problem. Appealing to the principle of economy of means, logical binary operators were rejected in favor of a general alternate mechanism which allows any compound operand or operator to be specified as an alternative.

Consider now the following four recipes:

```
 *pork *butter /sauté in *parsley /sprinkle with

 *beef *butter /sauté in *parsley /sprinkle with

 *pork *butter /sauté in *mint /sprinkle with

 *beef *butter /sauté in *mint /sprinkle with
```

One way of combining these into a single metarecipe would be to specify them as unrelated alternatives:

```
[ *pork *butter /sauté in *parsley /sprinkle with |
 *beef *butter /sauté in *parsley /sprinkle with |
 *pork *butter /sauté in *mint /sprinkle with |
 *beef *butter /sauté in *mint /sprinkle with ]
```

One really wants to express the family relationships among the four, however, by factoring out their commonality:

```
[ *pork | *beef] *butter /sauté in
[ *parsley | *mint] /sprinkle with
```

This is not only shorter and clearer, it is also a better expression of the general tree pattern which all four recipes share. Suppose, however, that instead of four recipes there were only two—pork with parsley and beef with mint. This requires the use of variants linked together by means of variant numbers:

```
[ *pork #1 | *beef #2] *butter /sauté in
[ *parsley #1 | *mint #2] /sprinkle with
```

A special kind of linked variants arises in the case of nominal variants, that is, variants which affect the name of the recipe. This case is treated by using the same notation in the title:

```
< *[PORK WITH PARSLEY #1 | BEEF WITH MINT #2].
[ *pork #1 | *beef #2] *butter /sauté in
[ *parsley #1 | *mint #2] /sprinkle with>
```

It should be noted that using nominal variants, the Metarecipe Generator can easily merge any two arbitrary recipes; the entirety of the world's cuisine could be represented as a single large recipe. The undesirability of doing so raises a number of issues in recipological taxonomy; we shall return to the subject in the section on suprarecipial organization.

3. Comments and annotations

Comments. We have claimed above that RxOL allows the controlled incorporation into the recipe of the auxiliary information which we argue can overwhelm traditional recipes. This is done through the following five forms of comments:

Modifiers. Adjectives and adverbs, applying to operands and operators, respectively, are simply placed after their argument using a comma separator:

```
*tomatoes, canned =chop, coarsely
```

It is through these that culinary techniques and information on culinary utensiles can be introduced. With ingredients the question often arises of whether to place an adjective after the noun as a comment or before it as an integral part of the ingredient specification. Should it be "green beans" or "beans, green"? "Olive oil" or "oil, olive"? The general rule is that if there is another form of the noun which could conceivably be substituted, then the adjective should be placed afterwards, as a comment.

Measurements. The units of measurement allowed are:

```
        ml mm g C kPa h min s
```

All Cocina software is committed to the metric system; as a concession to the American market, however, customary units can be automatically displayed as comments after the metric units.

Annotations. While modifiers and measurements are understood by the Cocina software tools, the remaining forms of comments are not; they provide information for humans only, and are simply ignored by the software.

Annotations are phrases attached to operands or operators, typically providing more details on culinary technique than a single modifier can provide:

```
=simmer, 30 min; should be nearly dry
```

Parentheses. When the information to be incorporated is not naturally attached to any particular node, the general-purpose mechanism of parentheses is the notation of choice.

```
(This sauce goes especially well with fatty fish.)
(An excellent hot-weather dish.)
```

Italics. For those who simply must have their chitchat, RxOL does allow one final form of comment. This is the preferred method of introducing material that is avowedly immaterial to the preparation of the dish.

```
My uncle Fred used to cook this when we went fishing together.
```

```
The Spanish influence on the Phillipines shows through in this dish.
```

```
 *tomatoes, large, ripe, 500 g (must be fresh)
```

4. Recipes

A recipe consists of:

```
 (i)    A start symbol, "<" (ii)="" a="" symbol="" to="" indicate=""
whether="" it="" is="" compound="" ingredient="" (="" *),="" unary=""
operation="" or="" binary="" ).="" obviously,="" most="" recipes="" are=""
ingredients.="" (iii)="" the="" name="" of="" recipe,="" optionally=""
including="" notes="" separated="" by="" semicolons.="" followed=""
period.="" (iv)="" reverse="" polish="" formula="" for="" recipe.="" (v)=""
any="" local="" subrecipes.="" (vi)="" recipe="" end="" symbol,="" '=""
class="jop-noMdConv">'.
```

5. Sequencing

One of the hardest points for the novice to understand is that a recipe's structure does not correspond to its chronology. On the surface, it would seem that what one wants is a step by step sequence of instructions for executing a recipe. It is the purpose of this section to explain why that is neither possible nor desirable.

In computer science terminology, the tree structure of a recipe imposes a partial ordering on its operations, but not a total ordering. All that is required by the recipe is that at the moment a binary operation is executed, both its operands must be ready; it says nothing about which of them was prepared first. Thus, given the tree:

```
 *lettuce =chop
 *parsley =mince /sprinkle with
```

it makes absolutely no difference whether the lettuce is chopped before the parsley is minced, or vice versa. Thus any recipe which specifies such an ordering, whether it be "Chop the lettuce, then mince the parsley. Sprinkle the lettuce with the parsley" or "Mince the parsley, then chop the lettuce. Sprinkle the lettuce with the parsley" is living a lie. This lie is so pervasive in "standard" recipes that it takes a long while for a cook, even an experienced one, to free himself from the notion that the order in which the operations are presented is not necessarily the order in which they should be executed.

This illusion would be relatively benign except for one thing. In order to minimize the preparation time of a recipe, the longest path from the leaves of the tree to its root must be started first. In computer terminology, this is the "critical path" of the recipe. Consider for example the following recipe:

```
*beef
*oil /sauté in
*peppers =mince /add =sauté
*mushrooms, dried =soak, 30 min /add
*spinach =chop /add
```

If the cook sautés the beef and peppers in oil, and then pauses for 30 minutes to soak the mushrooms, he will be in deep trouble. Not only will his stir-fry be soggy and greasy, but he will also have extended his total cooking time by the time needed to sauté the beef in oil.

In the case of a human cook, optimal performance can be ensured only by jealously preserving the freedom to prepare the operands of a binary operation in either order. This is because minimizing overall cooking time requires the greatest possible amount of parallel processing, which in turn requires, given the variability of the time it takes to perform any given operation and the large number of interruptions a recipe normally undergoes, that the cook be constantly scanning the recipe asking the question "what could I be doing now?" RxOL is designed to facilitate that bottom-up scanning process.

In the case of culinary robots, a much higher degree of parallelism will be possible, because there can truly be multiple processors. A robot cook is not restrained by the human inability to chop an onion, grate cheese, and sauté chicken all at the same time. On the other hand, the detection of parallelism in such a situation will have to be automated. The main obstacle to such automation is ingredient substitution: the precise sequence of instructions issued by the sequencer will be quite different depending on whether one is using dried garbanzos or canned ones, frozen peas or fresh ones, and so on.

Given such a sequencer, it would of course be easy to modify its output so as to be usable by humans, resulting in the interactive execution monitor discussed above. The overall performance of such a system is likely to be inferior to that of an experienced cook interpreting RxOL directly, but it might be easier to use, especially if several recipes are being prepared at once.

6. RxOL SYNTAX

```
NOTES ON NOTATION:

| denotes alternation
( ) are used for grouping
< > denote "zero or more occurrences"
[ ] denote "zero or one occurrence"
{ } denote "one or more occurrences"

file ::= {< ["§"|"¶"] string "."> {recipe}}
```

```
recipe ::= "<" block=""  ">"

block ::=
 " *" heading compound-operand |
 "=" heading compound-unary-operator |
 "/" heading compound-binary-operator

heading ::= phrase < ";" string > "."

variantlist ::= "#" integer < "," integer >

compound-operand ::=
 simple-operand |
 compound-operand simple-unary-operator |
 compound-operand compound-operand simple-binary-operator |
 variant-operand

compound-binary-operator ::=
 [compound-unary-operator] simple-binary-operator
 [compound-unary-operator] |
 variant-binary-operator

compound-unary-operator ::=
 simple-unary-operator |
 compound-unary-operator compound-unary-operator |
 compound-operand simple-binary-operator |
 variant-unary-operator

variant-operand ::=
 "[" compound-operand [variantlist]
 { "|" compound-operand [variantlist] } "]"

variant-binary-operator ::=
 "[" compound-binary-operator [variantlist]
 { "|" compound-binary-operator [variantlist] } "]"

variant-unary-operator ::=
 "[" compound-unary-operator [variantlist]
 < compound-unary-operator [variantlist] > "]"

varop ::="|" | "?"
```

```
simple-operand ::= " *" [":" | "!"] sentence

simple-unary-operator ::= "=" ["+"] sentence

simple-binary-operator ::= "/" ["+"] sentence

sentence ::= phrase < annotation >

annotation ::= ( "," | ";" ) string

phrase ::= idchar

string ::= title_char idchar ::= letter | "-" | """ | "'" | " "

title_char ::= idchar | digit | "[" | "]" | "#"

RULES:

1\. If any variant formula governed by a variant operator has a label, then
(a) all the variant formulas governed by that variant operator must be
labeled, and (b) the set of labels within the scope of that variant operator
must be equal to the set of labels in the heading.
2\. Comments are delimited by "(" and ")".
3\. Compound left-hand operands to binary operators may be bracketed using "
{" and "}'
4\. Blanks and comments are allowed everywhere except in numbers.
```

Chapter 4. An Introduction to Cocina.

1. General Information

Cocina is a set of culinary software tools which provide a variety of recipe selection and analysis functions designed to meet some of the goals set forth in Chapter 1.

Invoking Cocina. The icon for Cocina itself is a Macintosh with a baker's hat; double-clicking on that icon starts up Cocina with an untitled document. The icon for Cocina recipe files shows a plate and a wine glass; double-clicking on such an icon starts up Cocina and opens the selected document. Cocina libraries, which group together multiple recipe files, have icons depicting a pile of cookbooks, and may also be opened by double-clicking.

Cocina uses two auxiliary files, the Macintalk system file and a file called 'Ingredient Data'.

Editing windows. Cocina allows the user to open as many windows as available memory will allow. Each window is a text editing window supporting all the normal Macintosh editing features, including printing from the print menu. Many Cocina commands create new windows for the purpose of displaying output.

Cocina recipe files may optionally contain subtitle resources which provide titles for subsections of the file. These subtitle resources are created automatically when files are saved; any text between a section symbol ('§') and a period or end of line is treated as a subsection title, although only the first 15 characters are retained. When working with such files, the current subsection title will be displayed in the lower left corner of the window.

Files larger than 32K bytes may be opened for reading only. Editing commands which modify the content of the file are not available for such files; another editor such as QUED or Edit must be used to create or modify such large files.

Most of the Cocina commands described below operate on the current selection range, if it is not empty, or the entire file, if the selection range is empty. This applies to the print command as well, so that individual recipes can be printed without copying to another file. However, if the file is larger than 32K, a selection range must be chosen.

When a library has been opened, either by double-clicking on a library icon or by using the "open" command, a menu containing the files in the library will be available. Any file in the library may be opened simply by choosing its name from the menu.

2. File Commands

Add to Library. This command is used to add recipe files to the current library. The standard file dialog will appear repeatedly; selecting a recipe file will add it to the library. Only the name of the file is retained; if it is not in the current working directory, a utility such as "Set Paths" will have to be used to ensure that the file is accessible when it is selected from the library menu. Files may be deleted from the library by selecting them from the library menu while holding down the option key.

No provision is made for creating new libraries from within Cocina. Existing libraries may be copied and then modified, or new libraries may be created using a tool such as Menu Creator, Rmaker, or Resedit: a library is simply a file containing a MENU resource with an ID of 8.

3. Edit Commands.

Read Phonetic. The selected text is interpreted as text transcribed in the International Phonetic Alphabet, and is passed to Macintalk for speaking. An IPA font is provided for the purpose of entering such text. This option allows total control over the Macintalk output.

Translate to Phonetics. The selected text is translated to the International Phonetic Alphabet for processing by the Read Phonetic command. The algorithm used is rather crude; for the Beta release it is just the Macintalk Reader module. Future releases will have a translator specialized for culinary terms.

Sort. The selected text is sorted by lines. The primary purpose of this command is to allow sorting of shopping lists (see below). The algorithm used is sub-optimal, so performance on very large selections will suffer.

Select Random. This command chooses a recipe at random from the current file. The title of the recipe is highlighted. This command is very useful in putting together recipe lists, as the user is relieved of the burden of having to make selections. Holding down the option key when processing large files will restrict the choice to the current buffer. This speeds up response as well as limiting the scope of the search.

When a library has been opened and the current file belongs to it, the recipe will be chosen at random from among all the files in the library; the option key may by used to restrict the selection to the current file.

Dual Units and Go Metric. Because of its manifold advantages to the practicing cook who wants to get a meal on the table in the shortest possible time, and because Cocina is a long-range project, the metric system has been used throughout. Nevertheless, I realize that a substantial proportion of the Beta testers will be among the 2% of the world's population that is not familiar with the metric system. Consequently Cocina includes two commands carefully designed to make the system usable by the hold-outs while maintaining the dominance of the metric system. The Go Metric command converts customary units to metric ones, while the Dual Units command inserts customary units as comments after the metric ones. The goal of the conversion routines was readability rather than precision, so a pound is treated as 500 g, and so on. The results should be treated with care when dealing with baking recipes. The units which these two commands know about are:

Metric: g, m, J, Pa, l, °C

Customary: oz., lb., inch, kC, psi, t., T., cup, °F

The only two prefixes recognized are 'k' and 'm'.

Find. The bare-bones find command is case-insensitive. If a library is open, this command will automatically close the current recipe file and open the next one as needed. A replace command will be provided for the next release.

4. View Commands

The view commands operate on the selected text and transform it in one way or another to look at in from a different perspective. There are two groups of view commands: the first group presents the recipe itself in a variety of ways, while the second performs analyses and displays them, rather than the recipe itself.

If the selection range embraces more than one recipe, the selected command will be performed on each of them in turn. Typing - terminates this process.

Pretty Print. The pretty print command displays the recipe as a uniformly formated RxOL formula. It is useful to guarantee uniform formatting after a recipe has been entered. It inserts braces around compound right operands to improve readibility.

Natural Language. This command converts the selected recipes to natural language and displays them in the FNL window.

Tree. The Tree command displays the selected recipe as a tree diagram. This is useful in visualizing a recipe, in planning how to approach it. To avoid the problem of scrolling graphics, the Beta release version simply displays the tree on a full-screen window and waits for the user to click the mouse.

Nutrition. The Nutrition command displays a nutritional analysis of the recipe. Currently this analysis is limited to the recipe's food energy content, but will be expanded to other aspects in the near future.

Shopping List. The Shopping List command generates a list of the ingredients of the selected recipes, preceded by a category such as 'dairy' or 'produce'. By sorting the Shopping List window, a list conveniently arranged by category is produced.

Complexity. The Complexity command analyzes how complicated a recipe is. It does so by preparing a weighted sum of the number of ingredients and operations the recipe involves, with a penalty for recipes that are highly branching. In addition, this command computes the 'skeleton' of the recipe, that is, its essential structure disregarding straight-line series of 'add' and 'and' operations. Future releases will include an estimate of the time required to prepare the recipe.

5. Tool Commands

The Tool menu provides three analysis tools that are useful in planning an overall nutritional and exercise program. Each tool brings up a dialog window which accepts input from the user and calculates certain output values.

Exercise. The exercise calculator provides precise information about a given exercise regime. The three independent variables involved are body weight, altitude, and type of exercise (cycling, swimming, or running). Changing any of these values causes recomputation of all values based on the current distance and time.

The four dependent variables involved are distance, time, energy, aerobic points, and oxygen consumption. Changing any two of these causes recomputation of the remaining values based on the two new values.

For the sake of convenience, two additional variables are provided: speed and activity coefficient. These are just distance/time and energy/(weight.time), respectively.

To enter a value, edit the corresponding field and then click the "calculate" button.

Dieting. This window provides information on dieting strategies. Given an individual's sex, frame size, height, weight, age, food energy intake, and activity coefficient, along with a desired final weight, the tool computes the individual's ideal weight, the time it would take to reach the final weight given the indicated nutritional regime, and the food intake required to maintain both the current and the ideal weight.

Activities. This dialog allows the user to compute his activity coefficient by specifying the number of hours per day he spends on a variety of activities. A number of activities ranging from "lying still" to "running 15 km/h" are presented, along with the number of hours per day unaccounted for and the total activity coefficient. By filling in the boxes until the total reaches 24 (and the "unaccounted for" figure drops to 0), a customized estimate of an individual's activity coefficient may be derived. This figure can then be used in the Dieting Tool to estimate the time required for a weight change.

Three activity boxes are labeled "other", and have an editable field in which the user can fill in coefficients obtained from the Exercise Tool to allow for customized exercise regimes.

7. Getting Started with Cocina

The tools Cocina provides may be used in any number of combinations, but to get started we have found that the following scenario is quite useful.

1. Open an empty window called 'This Week's Menus' to store the recipes you will be preparing this week.

2. Open 'The World's Cuisine,' the master file of international recipes. It is from this file that you will select recipes.

3. Select a random recipe using the 'Select Random' or -J (for jump) command. If the recipe sounds good, copy it into This Week's Menus. If not, select another random recipe.

4. Continue in this fashion until a week's recipes have been chosen. Save your new file.

5. If desired, convert the recipes you have chosen into English by choosing the FNL command. In either case, print it out using the print command to use while cooking during the week.

6. With an empty selection range in This Week's Menus, choose Shopping List. This will generate an unsorted ingredient list in the Shopping List window.

7. Making sure the Shopping List is active, choose Sort from the Edit menu to arrange the shopping list by categories, then print it out to shop with.