Presenting dynamically expandable hypermedia

C.A. Macnee, W. Behrendt, J.R. Kalmus, K.G. Jeffery, M.D. Wilson
Rutherford Appleton Laboratory, Chilton, Didcot, Oxon, OX11 0QX, UK

Abstract

The Multimedia Information Presentation System (MIPS) will allow end-users to browse multimedia information presented in a user-friendly and consistent manner. In its most powerful configuration, it will allow the end-user to formulate queries which are interpreted, analysed, and despatched by the system to heterogeneous distributed external data sources, and to view a coherent and customized presentation of the data retrieved as answers. Data are stored in, or referenced from, a set of hyperdocuments conforming to the ISO standards HyTime and SGML. The hyperdocuments constitute an information web which may be dynamically expanded to accommodate retrieved data. The web navigation structure, structure of information nodes, specification of presentation mechanisms, specification of presentation tools, and data are separable and potentially reusable for different applications, different activities within an application, or different environments. We outline the intended functionality and the design of MIPS, with particular reference to the structure and function of the hypermedia web and the role of the knowledge base system module in its dynamic expansion.

Introduction

We outline the functionality and the design of the Multimedia Information Presentation System (MIPS), currently being developed by a European consortium. We make particular reference to the structure and function of the hypermedia navigation web which is constructed for an application of the system, and to the role of the knowledge base system module in the web's dynamic expansion. The paper has four main sections:

(1) the introduction, in which we describe the rationale for MIPS, the current capabilities of systems for retrieval and presentation of multimedia information, the MIPS demonstrator application being built for the domain of tourism, and the intended users and scenarios of usage for MIPS;

(2) a brief outline of the modular, client–server architecture of MIPS;

(3) a description of the structure and function of the hypermedia web at a conceptual level, and its syntax as defined by the MIPS Document Type Description conforming to the Hypermedia/Time-based Structuring Language (HyTime);

(4) a description of the operations carried out by the knowledge base system module with

respect to presentation of assets and expansion of the web.

Rationale for MIPS

It is currently possible with commercial products on a PC to retrieve information from databases and documents across a network whether that information be text, relational tables, still images, sound, or video. That information can be presented through tools in a commercial windowing system to users for them to read, or cut and paste into multimedia documents. Unfortunately, the range of different data sources which can be used is limited; queries must be specified separately for each of the data sources and not as a single query to retrieve the information from all of them; and the tools to present the information will each occupy a different window on the screen and use proprietary presentation styles which differ from each other depending on the source format of the information. The second currently available form of presenting multimedia information is by authoring it in a proprietary tool into a discrete document or hypermedia network which the user can then browse. However, the user has no access to documents outside the hypermedia network and is tied to a proprietary representation format.

The Multimedia Information Presentation System (MIPS) project seeks to combine the best of these two approaches and thereby to overcome the problems of each. MIPS will allow an end-user to retrieve and browse heterogeneous multimedia information structured and presented in a user-friendly and consistent manner. In its most powerful configuration (see the outline of scenarios of usage, below), it will allow the end-user to formulate queries which are interpreted, analysed, and despatched by the system to heterogeneous distributed external data sources, and to view a coherent and customized presentation of the data retrieved as answer. Prespecified or dynamically retrieved assets are stored in, or referenced from, a set of hyperdocuments conforming to the ISO standards HyTime[1,2] and SGML (ISO 8879). The hyperdocuments contain a navigation web of information nodes and hyperlinks. Each information node specifies presentable-data elements (which may be instantiated with data) and the means by which they may be presented as components of a GUI for the node. The data assets, web, and application-specific knowledge in a knowledge base system (KBS) module, together constitute the application description. This is authored by an application builder for a particular domain — e.g. tourism, aircraft maintenance, medical anatomy and surgery, ... The system uses the KBS (a) to mediate the interpretation and analysis of queries to data sources, (b) to act in lieu of an application builder in dynamically specifying fragments of hyperdocument, especially for the incorporation and presentation of retrieved data, and (c) to provide graceful adaptation to the needs and preferences of end-users and the constraints of the system configuration in use. In the present paper we describe (b) and (c). The approach we adopt in using a

KBS to mediate access to heterogeneous distributed data sources is discussed elsewhere.[3,4]

Current capabilities of systems for retrieval and presentation of multimedia information

To date, pre-authored multimedia applications have been immutable, stand-alone entities which incorporate both presentable assets and link structures in application-specific encodings. (A recent book[5] provides an impressively presented collection of images from available hypermedia systems, collected by graphic designers.) The application, in which data and presentation specifications are intimately incorporated, might typically be stored in a single CD-ROM. This is compatible with the concept of a media product — e.g. book, magazine, film — held by the conventional media community. The informatics community, on the other hand, think in terms of updatable data, often held in widely distributed sources in heterogeneous formats and accessible remotely. Prototype systems for presenting integrated multimedia documents over digital networks are under development. The DEMON system, for example, focuses on techniques to enable the presentation of high-quality multimedia over a limited-bandwidth network.[6] The introduction of broadband networks, such as SuperJANET in the UK, will allow development and testing of other prototype systems which need not be so constrained by network bandwidth considerations.[7] The World Wide Web and browsers recently developed for it are discussed below.

More fundamentally, the separation of data and control, and the consequent reusability of both data and control mechanisms, is of prime importance in computing. A recently developed system[8] addresses the problems of separating control and data for multimedia presentation. The Microcosm system is described as an open hypermedia system, which enables the construction and traversal of links from one multimedia document to another. The system enables easy construction of links into and out of applications that are not part of Microcosm. It creates hyperlink files, readable by the system, which index standard files that do not themselves therefore need to contain any markup tags readable by the system. MIPS also separates 'control' structures — navigation structure, node structure, specification of presentation mechanisms, specification of presentation tools — and data. (A presentation mechanism is defined as a set of window instances, plus any audio output, and a script of actions associated with windows or subwindows.) In MIPS, data may be incorporated in the web, stored in a local filestore, or accessed in remote data sources. Locally or remotely stored data can be used by the system without being recoded. Further, international standards — HyTime and SGML — are used for the specification of the 'control' structures. This is a significant advance — even for that scenario of MIPS usage, fixed presentation of fixed assets, which is equivalent to conventional pre-authored multimedia applications. In addition, the specification of presentation mechanisms is separate from the specification of presentation tools, easing portability to vary-

ing environments. Further still, node structures are separable from the specification of presentation mechanisms, thus facilitating variation in the modes of presentation of the data. Nodes are are also separable from navigation structure, which facilitates re-use of the contents of information nodes for different tasks within an application or for different applications. This contrasts with systems in which links are embedded in node structure (e.g. the hypertext system Hyperties[9]). The use of generic presentable-data elements in node structures eases reuse of those structures for other applications.

World Wide Web browsers

The advent of the World Wide Web on the Internet, and the development of browsing tools for it, such as Mosaic, has produced an enormous upsurge in interest in remote retrieval and presentation of multimedia information. Although WWW and its browsers have greatly facilitated the process, problems remain:

- *information location* — the serious problem of locating information on the Internet is improved by the WWW interface, but finding the required server is still very much a hit-or-miss process. The user has to know the location of the information s/he wants, and must access the site directly. MIPS will automatically locate data which is appropriate to a query and indexed in its library of information on external data sources.
- *information integration* — if data is retrieved from several sources on a topic, particularly from several DBMSs which supply tabular information as answers to queries, it needs to be integrated, and existing WWW clients make no provision for this. The user will receive a table of data from each data source, each displayed in its own format. MIPS will automatically integrate answers from multiple data sources and display a single, coherent presentation of the answer to the user's query.
- *ontology and query standards* — there are no clear standards for indexing information, thus exacerbating the problems of locating and integrating it. The user has to know or infer the access language for each site visited, and the terms used at each site to refer to the information required. MIPS will automatically translate the terms of a user's query into the terms used at each appropriate external data source in its library.

Demonstrator application

A demonstrator application is being developed for the domain of tourism; the application is envisaged to be available, in a travel agent's office, for the browsing and booking of packaged and non-packaged holidays. The MIPS presentations will combine the visual appeal of a holiday brochure, plus the advantages of video and audio output, with the availability of up-to-date information through database access. One example of the advantages offered by MIPS compared with the facilities conventionally available to a travel agent is the following. Poten-

tial purchasers of a holiday choose, from a brochure, a hotel in a certain resort. They find, when the travel agent makes enquiries, that it is fully booked. They may feel that those that are available are somehow inferior to their first choice, and they may therefore be less likely to make a booking. The MIPS application can be set to show only those hotels in the resort which have vacancies, thus preventing them from being stigmatized as 'second best' choices. The data retrieved by MIPS from appropriate external data sources shows the current state of the domain objects (hotels in this case) at the last update of the data sources. A holiday vendor, by comparison, might typically reprint its brochures every six months.

Users and scenarios of usage

Three scenarios of usage are envisaged for MIPS, as outlined below. The system functionality for each scenario includes that for preceding scenarios.

Scenario 1: Fixed presentation of fixed assets. All assets are already incorporated in the application description or stored locally. The presentation mechanism for each node is prespecified, as are the presentation tools to be used. The application is an immutable, stand-alone entity, and it might typically be stored on CD-ROM.

Scenario 2: Assets can be dynamically retrieved from external data sources in answer to prespecified queries. The application description contains queries, prespecified by the application builder, each of which is despatched if the end-user accesses the node containing it. The application description is expanded to incorporate or refer to (a) the data retrieved in answer to the query and (b) the specification of the mechanism of its presentation. The fragment of web which is to accommodate the retrieved assets, and the mechanism and tools for presenting it, may be prespecified by the application builder or dynamically specified by the KBS.

Scenario 3: Assets can be dynamically retrieved from external data sources in answer to user-generated queries. The user can, at any juncture in navigation, formulate a query to external data sources. The query may be formulated by completing a query template or by construction of an open query using a constrained vocabulary. An appropriate web fragment and presentation mechanism for the answer are specified dynamically by the KBS.

In order to allow the flexibility required of MIPS, three different classes of user are envisaged at different stages in the development and use of an application.

- First, an *application builder* is required to author the initial HyTime hyperdocument containing the link and node structure, provide data assets to be presented or identify sources from which they can be obtained, and enter domain knowledge into the MIPS KBS. MIPS does not provide HyTime authoring tools, but it is predicted that the suppliers of existing

multimedia authoring tools will provide filters from their proprietary representation to this ISO standard to enable interchange of documents, as currently do word processors to the SGML standard on which HyTime is based. Application builders would therefore use the facilities available in their preferred authoring tools, without the overhead of learning new ones specific to MIPS. Data assets of text, static images, or video could be produced by application builders in their preferred tools, or used from existing sources provided by third parties. General KBS domain knowledge will already exist in MIPS for the application builder to expand in the application domain to enter details of data sources from which presentable information can be retrieved. If the application domain is one in which previous applications have already been produced (e.g. tourism) then this would only require very minor development.

- The second user of the system will be a *site configuration manager* who will install the MIPS application and further update the KBS with details of the presentation tools available at the site and the exact class of end-users who will use the system.
- The third class of user comprises the end-users who will be presented with the assets represented in the hyperdocuments. They will obviously vary in their experience of the domain and in using a MIPS application. To support this variety of end-users, the KBS includes detailed user models which are used to select appropriate presentation mechanisms and tools.

Each of these users has a role to play in the progressive tailoring of a MIPS application. The application will be tailored to a domain, site, and finally to the actual end-user at that site, in order to provide the most effective and efficient presentation of information.

The design of an embedded user modelling facility entails a trade-off between the expressiveness and coverage of a natural language interface which actively infers the user's beliefs, on the one hand, and the run-time efficiency of a tunable defaults file approach, on the other.[10] The goal of the engineering solution embodied in the KBS's user modelling system is to have the required information available while the user is interacting with the system for the interface options to be set in the most cooperative way for users to achieve their task objectives. The addition of a monitoring function will allow the system to be auto-adaptive to the user: the system will monitor the progress of a user on the attribute which indicates that they have reached the stage of experience where they would change the interface, and change it for them at that time. The development process includes: identifying the variations in the user population; identifying the variations in the interface/application functionality; identifying correspondences between classes of user (stereotypes) and the required interface functionality; identifying trigger or cue events which determine that a user has changed a user class (macro rules); representing this correlation as efficiently as possible in a user model; embedding the

user model in a user modelling component which eavesdrops on the user's actions. At run-time, the user's actions are monitored for cue events or combinations of cue events. When events occur, the user modelling component will trigger macro rules and change the user stereotype which will cause the customizable user interface or application options to change when the user modelling module is queried for the values of these options during run time.

Architecture

In order to control the complexity inherent in the system and to promote interchangeability of presentation tools, the architecture of MIPS is based on client–server relationships between modules.[11] The architecture is illustrated in Figure 1. The major modules are as follows.

[Figure 1 near here]

- The *Presentation Manager* (PM) exerts overall control of the system, in response to instructions contained in the application description or events arising from user interaction. In addition to calling other modules as described below, it activates presentation tools to display assets to the user. Where all or part of the presentation mechanism for a node is unspecified by the application builder, the PM calls the KBS to provide specifications customized to the current user.
- The *HyTime Engine* is a library of functions to manipulate HyTime and SGML documents. A server to the PM, it executes the creation, modification, and deletion of (fragments of) such documents held in the *HyTime Store*, as well as interrogation and navigation within a document. It also enables the import and export of such fragments to and from the HyTime Store. The functions are defined according to the ISO standards, independently of MIPS. The HyTime Store is built above an object-oriented repository, since the abstract structure of an SGML or HyTime document is a tree or a general graph.
- The *Selection and Retrieval Tool* (SRT) is called by the Web Builder to process queries to external data sources. It decomposes the query into subqueries suitable for despatch to the data sources identified by the KBS as appropriate. It may call the General Query Tool to enable clarification of the query through a dialogue with the end-user. It arranges for the subqueries to be despatched to external data sources by a *Communications* module. It assembles the data retrieved in response to subqueries into a consolidated answer and sends it to the Web Builder.
- The *Web Builder* (WB) is called by the PM to expand the web if the latter encounters in the web: a template which requires to be instantiated; a query which is to be despatched to external data sources; or an instruction to call the GQT. In the case of query processing,

the WB calls the SRT, which may then call the GQT. The WB calls the KBS to find or create templates suitable (a) for the specification of a fragment of web — navigation structure or node structure as necessary — to incorporate or refer to the consolidated answer received from the SRT and (b) for the specification of appropriate presentation mechanism(s).

- The *Knowledge Base System* (KBS) is the application's source of knowledge about the application domain and the tasks that users will want to carry out over it. It also maintains models of users of the application. It serves the PM by customizing presentation mechanisms to the preferences of the user (a) by setting general parameters for a session, based on its model of the current user, or (b) when providing templates for some part or all of the presentation mechanism at a node. Acting in lieu of the Application Builder, it serves the WB by providing templates for the dynamic construction of fragments of web. It serves the SRT by guiding the decomposition of queries and identifying appropriate external data sources. It also provides the SRT with schemas of the expected and actual answers to queries, to guide the assembly of the consolidated answer and the selection or construction of web templates. It serves the General Query Tool by providing query templates and managing dialogue with the end-user.
- The *General Query Tool* (GQT) is called by the SRT to provide the interface to enable a user to clarify or formulate a query. It calls on the KBS as described above. The finished query is passed to the SRT for further processing.

The presentation tools and GQT will interface with the window management system for the configured application.

The number of modules required varies with the scenario of usage. The Presentation Manager, HyTime Engine and Store, a set of presentation tools, and a window management system will be necessary for all scenarios. For Scenario 1 the Web Builder will be required if templates are used in the web by the application builder. For Scenario 2, the Web Builder, Selection and Retrieval Tool, KBS, and Communications module are required; the GQT is required when it may be necessary to refine a query in order to reduce the amount of data retrieved. All modules are required for Scenario 3. The functionality described in this paper is that for Scenario 3.

Communication between modules is managed through interfaces which use a communications bus similar to RPC. The interfaces surrounding the Presentation Manager pass data structures related to the HyTime language. The interfaces around the Selection and Retrieval Tool for querying data sources and returning data pass data structures in an Internal Representation Language (IRL). The IRL performs the same task between submodules within the SRT.

Two aspects of the IRL are important in this respect: an applications communication protocol, which remains constant across the architecture and provides an envelope for carrying messages in different forms; and a message content, which supports the functionality of the particular module and submodule interfaces involved. The interfaces fall into three families, each relevant to one of three areas of functionality:

- breakdown and clarification of queries into sets of subqueries;
- interpretation and manipulation of retrieved data;
- the server role of the KBS throughout the architecture.

MIPS hypermedia documents — the HyTime web

The application description

The assets presentable by a MIPS are represented within a set of hyperdocuments conforming to the ISO standards HyTime and SGML. In the set of hyperdocuments is implemented a network of information nodes connected by hyperlinks, known as the HyTime web. The web and the application-specific knowledge contained in the KBS module, both of which are authored by the application builder, together constitute the application description. The application description defines

- the possibilities for navigation among nodes,
- the information presentable at each node,
- the modes of presentation of that information, and (possibly) the tools to be used to present it,
- the events associated with the user's interaction with the GUI generated for the node.

Conceptual structure of the web

The conceptual structure of the web is created by the application builder to meet user requirements for a particular application. In so doing, the application builder should be aware of the ontology of the application domain, the subset of object classes about which the user is likely to require information for any given activity within that domain, and the ways in which these object classes are related to one another. The types of nodes required, and the patterns of hyperlinks among them, can then be designed so as to facilitate browsing of the web by the user. The conceptual structure of the web will be related to the domain ontology constructed for the KBS module, and it may be possible to create different conceptual types of hyperlink to reflect different types of relation between objects in the ontology.

A simple example of conceptual web structure, for the domain of tourism, is one in which nodes having the application-specific types 'country', 'region', 'city', and 'hotel' are con-

nected in a tree where the links have the conceptual meaning 'part-of'. In the domain of aircraft maintenance, a large and complex web of nodes representing components might be organized so as to be browsable by structure of the aircraft (probably as a tree where the links have the meaning 'part-of'), or by 'walk-through' sequence of connectivity (where the links have the meaning 'gives-access-to'), or by functional system or maintenance cycle (possibly as a general graph where the links represent the functional roles of components or the sequence in which they should be inspected at a given maintenance period). The information contained in the nodes would thus be reusable for different activities within the domain, with each activity mapping to a certain conceptual type of hyperlink.

Navigation

At each node in the web, the user will be presented with a set of window instances (plus audio output if any) which convey

(a) (a subset of) the information which constitutes the contents of the node and

(b) the options available for traversal from the current node, in the form of a *preview* of the accessible nodes.

The preview will typically be presented as an interaction mechanism ('widget') showing the names, or some other brief descriptions, of the accessible nodes and allowing the user to select one for traversal. On selection of an accessible node, the contents and preview information for that node will be presented, and so on. Where the KBS mediates presentation of preview information, it may constrain that presentation according to the characteristics and preferences of the user. In a large and complex hyperdocument, this will help the user to avoid information overload and to minimize the risk of becoming 'lost in hyperspace'. Where different conceptual types of link are available, the preview information might be parameterized by type of hyperlink.

Templates

The MIPS approach to the construction of application descriptions, both by the application builder and dynamically by the KBS, involves the use of templates. Templates specify fragments of information web in a range of granularity from navigation structure to the basic components of information nodes. They may refer to other templates, so that high-level templates may be constructed from lower-level 'building blocks'. The rationale for the use of templates is twofold: to minimize the size of the application description by allowing reuse of structures specifying uniform presentation formats for an arbitrary number of instances of the same type, and to ease the task of the application builder (or the KBS) in creating the application descrip tion. We identify two principal categories of template:

- world knowledge type:

— application-domain-specific type
— generic world knowledge type;
- presentation type.

A template representing an application-domain-specific type, for example 'hotel' in the domain of tourism, will specify a uniform web-fragment structure and presentation format for instances of that type. The target-type of such a template, which is the type of web fragment which will result when the template is instantiated with data, will typically be an information node. The template may refer to 'building block' templates whose types belong to any of the above-mentioned (sub)categories. It is expected that there will be a partial mapping from the inheritance hierarchy of world knowledge types, as expressed in the KBS's ontology, to an inheritance hierarchy of presentation formats as specified by the corresponding templates. Templates representing types may specify (some of) the same interaction mechanisms and presentation attributes as do templates representing their subtypes. The use of uniform presentation formats for instances of the same type, and inheritance of aspects of presentation formats from types to subtypes, is expected to help the user to maintain orientation in a complex information web: i.e. it will be an associative aid to prevent the user from becoming 'lost in hyperspace'. In addition, depth in the inheritance hierarchy will be one dimension on which a best-fit match may be made by the KBS between (a) a schema characterizing the answer to a query to external data sources and (b) schemas characterizing web templates. Such 'fuzzy matching' will allow graceful degradation of the fit between retrieved data and presentation mechanisms. Examples of application-domain-specific types and their subtypes for which inheritance of attributes might map to inheritance of node structure and presentation format are:

- for the domain of tourism:
  — journey (subtypes: flight, rail_journey, road_journey, sea_journey, ...),
  — accommodation (subtypes: hotel, self-catering, pension, camping, ...),
  — activity (subtypes: entertainment, sport, aesthetic/cultural, ...);
- for the domain of aircraft maintenance:
  — mechanical component (subtype e.g. hydraulic component, subsubtype e.g. valve, ...),
  — electrical component ... and so on;
- for the domain of medical anatomy:
  — bloodvessel ...,
  — joint ...,
  — gland ... and so on.

One can imagine that the presentation format for all mechanical components will have the same background colour, type style and so on. A template for accommodation will share many interaction mechanisms with one for hotel, for example, but will not have a field for 'star-rat ing'. If, for some reason, no template of type hotel were available to accommodate retrieved data about a hotel, then the KBS could make do with a template of type accommodation, los ing only small subset of the data. Alternatively, the KBS may decide that the runtime cost of constructing a new template 'custom tailored' to the retrieved data is acceptable. Such a deci sion hinges on the trade-offs involved in tailoring presentation to suit data versus tailoring data to suit presentation.

The advantages of templates w.r.t. 'off the peg' matching and modular construction are still more evident in the case of generic world knowledge types, although the templates themselves are generally more difficult to conceptualize. Such templates would be readily portable across application domains. They may refer to other templates of the same type category or of the presentation type category, but not to application-domain-dependent type templates. An example type is 'amount', one of whose subtypes is 'cost'. A template of type 'amount', conceptualized at a fairly basic level, may simply specify the combination of a number and a string, whereas one of type cost might specify the combination of a real number and a string representing a unit of currency.

Templates of presentation type may be referred to by world knowledge type templates but not vice versa. An example is a template whose type is 'figure'. It specifies a compound interaction mechanism comprising a picture and a legend; the legend may be a title or a caption, and the picture may be static or interactive. It may refer to templates of type 'picture' and 'legend' and specify relative position: legend above the picture if the former is a title, below if it is a caption. The 'legend' template may refer to a template of type 'caption' or one of type 'title', and these in turn will specify different styles of presentation of a string.

Syntactical structure of web and representation of data

The syntactical structure of the web is expressed in terms of HyTime/SGML elements conforming to a Document Type Definition (DTD) for MIPS.[12] The DTD specifies the kinds of elements, and their attributes, that can be used to describe the structure of the web, and constrains the relationships among elements. (In other words, the MIPS DTD specifies the vocabulary and grammar to which a web, by analogy with a sentence, must conform. HyTime/ SGML may be seen as a meta-language which defines the grammar.) The syntax of the components of the web as specified by the DTD may be represented by a tree in which types of web fragment are related to their children by containing or referring to them (see Figure 2).

The root of this tree represents the entire navigation structure, and the leaves are either empty *presentable-data* elements or raw data. Raw data acquires MIPS-specific semantics by being delimited by mark-up tags bearing the generic identifier of an element defined in the DTD. Presentable-data elements may either have raw data embedded within them or make reference, from one of their attributes, to a primitive element (e.g. a list of items or a relational table) containing raw data.

[Figure 2 near here]

Raw data may be represented in an information node (a) directly, by inclusion as the contents of a presentable-data element (for small quantities); (b) by reference from a presentable-data element, typically via a template, to the address of a relational table elsewhere in the set of hyperdocuments; (c) by reference from a presentable-data element to the address of an item of local system storage; (d) implicitly, by including the specification of a query to be despatched to external data sources. Queries to external data sources may also be generated through the interaction of the user and the GQT. To accommodate data retrieved from external data sources, the web is expanded by the Web Builder, with the assistance of the KBS, either (1) according to the exact prespecifications of the application builder, (2) by the KBS's finding the best fit of the data to the available 'off-the-peg' specifications, or (3) by the KBS's acting in place of the application builder to specify 'custom-tailored' the composition of a fragment of web in which the retrieved data can be represented. The web fragment may vary in complexity from a presentable-data element in a node, up to a set of nodes and associated hyperlinks.

Hyperlinks and link-end structures

The general-purpose hyperlink in MIPS — the ilink element — conforms to the HyTime form 'independent link', and its anchors are information nodes. It may have an arbitrary number of link-ends, which allow access to the nodes by reference. (Alternatively, the nodes may be directly contained in, or referred to from, the contents of the ilink element, rather than from the linkends attribute.) Each node may be accessible from an arbitrary number of hyperlinks. In general, the traversal rules for independent links in HyTime allow the following distinctions: (a) traversal to another node accessible from the current hyperlink, versus traversal to a node inaccessible from the current hyperlink (but, of course, accessible from another hyperlink which addresses the current node); (b) initiation of a traversal from a node, versus return from a node to another from which it has been accessed. Allowable traversals from a node are specified by combinations of allowable (a) internal/external and (b) initiation/return rules.

In addition to HyTime-defined attributes specifying link-ends and traversal rules, a number of MIPS-specific attributes of an ilink element specify objects associated with each link-end and hence, implicitly, with the node to which the link-end refers. The objects are:

- an *auxiliary node* which provides a brief description (perhaps just a name) of the node, to be used when presenting a preview of the nodes accessible from a given node;
- a *presenter* template which, when applied to the node, constructs a presentation mechanism for the contents of the node;
- a *previewer* template which, when applied to the auxiliary nodes of the nodes accessible from the current one, constructs an interactive presentation mechanism for the preview;
- a *combiner* template which creates a presentation mechanism that combines those for the contents and preview for the node, including a script of actions. The actions are instructions to the Presentation Manager module: e.g. create a window instance, show a window instance, bind data to a window instance, get an item selected by the end-user, ...

Information nodes

Node elements are tree structures comprising presentable-data elements indexed by keys. There are three principal presentable-data element types:

- *enum*, in which a list of items (tokens) is keyed either to labels (strings) or the specification of hotspots;
- *fileloc*, which addresses a file in local system storage;
- *view*, which selects a subset of raw data from, for example, a relational table or spreadsheet.

Each node is characterized by an application-specific type name and by a *node schema* (see below).

The presentation mechanism for a node is specified by the combiner template of the link-end of the hyperlink from which it is accessed. The presenter and previewer templates to which it refers specify a GUI for the node as follows.

(1) For each presentable-data element in the node, or for the list of descriptions of accessible nodes, an *interaction mechanism* is specified: e.g. scrolling list, menu bar, buttons, boxes, interactive picture, interactive text, table, bar-chart, picture, audio, video, ... (Note that an 'interaction mechanism' need not necessarily allow interaction with the Presentation Manager.)

(2) For the node, the GUI is specified as a set of window instances. Within a window to be rendered by the dedicated MIPS Presentation Tool (MPT), subwindows may be specified as a tree of *frames*. The leaves of the tree of windows and frames are the interaction

mechanisms specified in (1). The layout of the set of windows, and of frames within a window rendered by the MPT, are specified either as a coarse description of relative position (above/below or left/right juxtaposition) or in terms of coordinates specified relative to a window or frame. Any temporal relationships among the windows — i.e. any staggered presentation — may also be specified (see 'Temporal issues', below).

The combiner template specifies the combination of the node contents window(s) and the preview interaction mechanism, plus a script of actions which provide instructions to the Presentation Manager (PM) on the presentation of the window(s) and on the events associated with user interaction.

KBS operations related to presentation of assets and expansion of the web

MIPS schemas

The KBS, acting as an 'automatic author', is called upon by the Presentation Manager to provide 'presentation' web templates to specify the presentation format for the contents of information nodes, and by the Web Builder to provide 'navigation and node structure' web templates (plus node schemas if appropriate) specifying those aspects of web fragments to accommodate retrieved data. Rather than attempting to dynamically match or construct templates on the basis of data itself, the KBS is provided with schematic representations of the structure and content of the data (or, failing that, it may have to construct them). There are three major types of schema: answer schema, web template argument schema (WTAS), and node schema. They specify the structural interrelationship of raw or presentable data components and characterize those components in terms of types in the KBS ontology.

Answer schema

The answer schema is the vehicle for information about data passed between the query processing and data retrieval/consolidation submodules (SRT) on the one hand, and the web building and presentation modules on the other. That information, where it is specified by the application builder, is interpreted and translated by the KBS; where it is absent, the KBS, acting in lieu of the application builder, has to infer it from the query. The application builder specifies the answer schema to impose structure on retrieved data. The purpose of that structure is to specify the components which are to be added to the information web to accommodate the retrieved data, and the way these components are interrelated in a tree; ultimately, that structure is manifest in the way in which data items are grouped together into hyperlinked, interactive screen presentations. SRT may also use the answer schema to 'collate' the retrieved data. Before retrieval of data, the structure is an *expected* answer schema (EAS);

after retrieval of data, and possibly amendment of the structure by the KBS to take account of uninstantiated components, it is an *actual* answer schema (AAS). The AAS, which is a model of the answer, is used as the basis for matching against WTASs (see below) or, failing a match, construction of a web template.

Web template argument schema (WTAS)

This characterizes the arguments of a web template in the same way that an actual answer schema characterizes the data retrieved in answer to a query. It is isomorphic with an answer schema.

Node schema

This may be regarded as a special case of the aforementioned schemas, for which the target-type is a node, generic presentable-data elements are specified, and GUI layout instructions may be specified.

Expansion of web to specify presentation mechanism

If one or more of the presenter, previewer, or combiner templates for a node is found by the PM to be missing when it accesses a node, then the PM will call the KBS to provide one. On the basis of the information contained in the node schema, the KBS constructs an appropriate presenter template. On the basis of a list of (sets of) descriptions of the auxiliary nodes for accessible anchors, the KBS constructs a previewer template. The KBS then constructs a combiner template including the specification of a script of actions.

The categories of knowledge which the KBS must possess in order to fulfill its role as an automatic author in this respect concern:

- the syntax of the MIPS HyTime DTD;
- file formats and data types of multimedia assets, and the corresponding input and output capabilities of presentation tools and convertors;
- algorithms and heuristics about the matching of assets to the capabilities of presentation tools and convertors;
- the range of interaction mechanisms which can be rendered by MIPS, either through the MPT or by launching a proprietary presentation tool;
- the set of interaction mechanisms which is appropriate to each type of presentable-data element;
- user characteristics;
- heuristics about optimizing selection and customization of interaction mechanisms

according to user characteristics and preferences;

- algorithms and heuristics about GUI design
  - — algorithms to allocate areas of the screen to the assets to be displayed in a presentation mechanism,
  - — heuristics about HCI/ergonomic factors and avoidance of information overload,
  - — heuristics about aesthetic factors;
- heuristics about selective presentation of the information available in a hyperdocument information web, according to user task or user characteristics,
  - — by constraining type or amount of information presented at each node:
    - – selecting, from the presentable data elements of an information node, those which are to be included in the current presentation mechanism for that node
    - – selecting, from those presentable data elements included in the current presentation mechanism, those which are to be 'unmapped' but displayable through interaction,
  - — by constraining navigation options:
    - – by hyperlink conceptual type
    - – by node conceptual type;
- the syntax of node schemas.

To construct the previewer template, the KBS is supplied with a list of contents of auxiliary nodes of accessible anchors. Where different conceptual types of link are available, the preview information might be parameterized by type of hyperlink, either by displaying an interaction mechanism for each type or by presenting the information in two stages: (i) the conceptual types of link along which a traversal can be made from the current node; (ii) on selection of a type of link, the nodes accessible by that type. Alternatively, the KBS may constrain the preview to the navigation options available by one type of hyperlink, according to its model of the current user. For example, in the tourism domain, if two types of link are available — general information and historical information — and the user is a travel agent rather than a client, the KBS may decide not to present historical information. The KBS also selects the form in which each destination anchor will be displayed in a preview widget, from those available for each anchor: e.g. string or icon. It then selects an appropriate widget.

Temporal issues

We distinguish three types of temporal issue in multimedia presentation, as follows.

(1) Synchronization of components of a 'canned' continuous asset — typically audio soundtrack to video — stored locally. This is handled in MIPS by passing control to the pres-

entation tool.

(2) Scheduling of presentation (without user interaction) of different asset types, or of different instances of the same asset type — e.g. a sequence of still graphics — stored locally. MIPS HyTime DTD has facilities for describing such schedules, using HyTime's Finite Coordinate Space module. This sequence might be scheduled to start at the same time as a video sequence, say, but would not be fully synchronized with it. The HyTime standard itself does not specify a scheduling algorithm; an automatic scheduler for multimedia document events is described, and compared with others, in a recent paper.[13]

(3) Real-time coordination of retrieval and presentation of different asset types (or of different instances of one asset type) from external data sources, to meet a predefined schedule — e.g. a piece of music, or a sequence of still graphics, to accompany a video sequence. We feel that this is not at present a practical proposition for MIPS; it would require very high bandwidth communications lines. Material stored externally will have to be pre-retrieved and cached before initiation of any temporal schedule.

An interesting issue for the future concerns the possibility of dynamic selection, editing, and temporal combination of different assets: e.g. the identification of a suitable piece of music of a certain duration to accompany a video sequence. Ideally, we want some method of tagging continuous assets by duration and by content (see remarks in the Conclusion) to allow for full synchronization.

Selection of presentation tools

When the presentation mechanism for a node is specified, the PM needs to activate appropriate presentation tools (PTs) to present its components. The PT to be used for each component may be pre-specified by the application builder or configurer of the system. The default presentation tool is the MPT. Where an interaction mechanism cannot be presented by the MPT, and a proprietary PT has not been specified by the application builder, the KBS is called to select an appropriate tool, and associated chains of file format convertors as necessary, from those available to the system. The KBS will minimize the number of conversions required.

The KBS infers the property values to be set for each tool (e.g. font size, background colour) from its user and context models.[14] We assume accessible APIs which allow us to specify these settings. The KBS then communicates the calls of the tools, including the specification of the settings, and the identities of the components of the presentation mechanism to which each applies, to the PM.

Expansion of the web to accommodate retrieved data

The full power of MIPS is manifest when the user can retrieve data from heterogeneous distributed external data sources and view a customized presentation of that data. To accommodate retrieved data, whether the query was (a) prespecified by the application builder and stored in the web or (b) generated through user interaction with the GQT, requires that the web be dynamically expanded during the user's session. This is done by the Web Builder (WB) module according to a specification supplied by the KBS in the form of a web template whose instantiation with the retrieved data will result in an appropriate fragment being incorporated in the web.

The web-expansion options for prespecified queries are:
(1) that the contents of the node from which the query was despatched are augmented to accommodate the answer; in this case the augmented contents of the node will be presented along with a prespecified preview;
(2) that a new node is created for the answer and linked to the 'query node' alone; in this case the unaugmented contents will be presented along with a preview including an option to access the 'answer node';
(3) that a new or updated navigation structure is created. This case may involve (a) the updating of the contents of existing nodes, (b) the deletion of some nodes, (c) the creation of new nodes of certain conceptual types, and the creation of links appropriate to those types.

The categories of knowledge which the KBS must possess in order to fulfill its role as an automatic author in this respect concern:

- the syntax of the MIPS HyTime DTD;
- heuristics about the design of hyperdocument information webs
  - — at the level of a navigation structure, i.e. conceptual typing of links, conceptual typing and instance-naming of anchors (information nodes), and setting of traversal rules to constrain order in which anchors can be accessed,
  - — in terms of allocation of assets to one or more information nodes,
  - — in terms of relationships between information nodes' presentation formats and their conceptual types;
- the syntax of answer schemas.

The KBS also has to amend the existing node schema for any augmented or updated node, or create one for any new node.

For GQT-generated queries, we envisage that the option of creating a new answer node will be chosen. The option to generate a query through the GQT will be provided at every node. This is likely to entail a 'superlink' to a 'GQT node' so that the latter can be accessed from all nodes in the web. (Any new node created by the WB would have to be added to the set of anchors of this hyperlink.) The KBS may therefore have to decide the node(s) to which the 'answer node' should be linked. We envisage that the default option will be to link the answer node to the GQT node alone. The GQT node will maintain a stack of the auxiliary nodes for the nodes which have been created during a session, and will provide these as preview of accessible nodes. (The traversal rules of the 'superlink' will ensure that nodes in the web at the start of the session cannot be accessed from the GQT node. The alternative, i.e. to make all links to the GQT bidirectional, would entail the user's being presented at the GQT node with a preview of every node in the web, thus maximizing the likelihood of becoming 'lost in hyper-space'. It will be possible only to access a dynamically created node, return to the GQT node, then return to the node from which the GQT node was accessed.) The user will access the desired answer node to be presented with the retrieved data.

Where no answer schema has been specified for a query by the application builder, perhaps because the query has been generated at runtime through the GQT, the KBS must construct one to the best of its ability on the basis of the query. This is a knowledge-rich task which involves the interpretation of the IRL(QD) expression of the query and the application of heuristics to group together the components of a query into nested relational tables which map to the syntax of the HyTime DTD. In addition to several of the categories of knowledge required for the construction of templates on the basis of node or answer schemas, the following are required for this operation:

- the syntax of the IRL Query Dialect (IRL(QD));
- application-domain-specific heuristics about the tasks to be performed by the user: component subtasks, priorities, information requirements;
- heuristics about single or multiple instantiation of query variables by retrieved data;
- heuristics derived from constraints of the syntax of the HyTime DTD;
- meta-typing in the KBS's ontology.

Mechanics of web expansion

An answer to a query is presented to the WB in the form of a set of (possibly nested) relational tables of raw data, known as the Consolidated Answer. Each relational table is characterized by a schema, and the top-level schema is known as the Actual Answer Schema (AAS). The AAS describes the components of the Consolidated Answer in terms of the typing information

contained in the KBS's ontology (as for a node schema, with the exception of MIPS DTD presentable-data element). The KBS may derive the raw data type by consulting its library of data about external data sources.

This answer schema is supplied to the KBS along with a *target-type*, which identifies the type of web fragment which is to accommodate the data contained in the Consolidated Answer. The KBS is also supplied, by BACA, with information on the number of data items instantiating each query variable, and with the original full query formula. The KBS uses the information supplied to it in two ways, as follows.

(1) The KBS seeks to match the Consolidated Answer to an existing template in the application description, whose instantiation with the Consolidated Answer as a set of arguments (actual parameters) will result in the specified target type. The matching process allows templates to be reused to accommodate a variety of answers (particularly to GQT-generated queries, for which no prespecified template exists); this is likely to be more efficient than constructing a template in each case. To be exact, the matching is attempted between the AAS and the WTAS for each template registered with the KBS. The WTAS may be considered to specify the conditions for selecting the template for instantiation by the data contained in the Consolidated Answer. Moreover, a degree of flexibility can be introduced in the matching process, so that the best fit of the answer to the available templates is allowed, within limits. There are two possibilities:

(a) an exact match is made (most likely with a template constructed by the application builder specifically for the answer to the query);

(b) no exact match is made. In that case, the KBS undertakes a constraint relaxation exercise to find the best match.

The criteria of fit for the match take account of: (i) the target-type and (ii) number of fields matched between the AAS and WTAS and the degree to which they match. The criteria of fit between fields relate to the degree of detail and the level of generality of the description of their contents. The more general and less detailed the description, the greater are the number of options for presenting the data, but the less well tailored will the presentation be to the particular application domain, the activity within that domain, and preferences of the user. To match, the WTAS field needs to be equally or less detailed and equally or more general than the AAS field, and, for the best fit, we seek the most detailed and least general WTAS field which matches. The process is analogous to choosing 'off the peg' a suit (the template) with which to present one's body (the Consolidated Answer) to the world. The suit has to be loose enough to accommodate the body, but one seeks the tightest fit that meets that criterion.

(2) if no acceptable match is made, the KBS constructs a template 'tailor-made' to the consolidated answer, on the basis of the AAS and the target-type. The principal steps in the process can be summarized as follows, for a simple case where the answer is to be incorporated in the web as a separate node linked to the node from which a prespecified query was despatched.

- For each field in the AAS, derive from the typing information one of the options available as children of the top-level target-type. In this case, the top-level target-type is a node and the children are presentable-data elements indexed by keys. The field name becomes the key, and a presentable-data element is inferred from heuristics. The process is repeated recursively for nested schemas and their target-types. At each level in the nesting, a 'building block' flexible template is identified which refers to templates specifying lower-level structures as necessary.
- A node schema is created from the AAS by identifying the appropriate presentation-data element for each field.
- A conceptual type is assigned to the node. This may be specified with the target type or may be inferrable from the application-specific typing in the AAS.
- Presentation templates are constructed to specify a presentation mechanism for the node, on the basis of the node schema. The KBS infers an interaction mechanism for each presentable-data element, on the basis of: the presentable-data element; the data format(s) of components, or the media-type for a fileloc element; user characteristics or preferences; capabilities of tools in the configuration to render interaction mechanisms.
- Heuristics for the composition and layout of windows are used to infer, from the set of interaction mechanisms for the node, the structure of a presenter template specifying the set of window instances to be used for presenting the answer.
- The construction of the previewer template is straightforward for this simple example, since there is only one option for navigation from the 'answer node': return to the 'query node'.
- A script of actions to which the windows and subwindows are bound is created, and incorporated in the design of the combiner template.
- A template specifying the new link is created, in which the appropriate link-end structures refer to the newly created presentation templates.

The KBS later assigns proprietary presentation tools to windows if necessary.

Conclusion

A prototype MIPS system for the domain of tourism is currently being implemented.[15] (Figure 3 shows a typical GUI design.) The design of MIPS allows not only for the development of hypermedia titles by authors and publishers, but opens up those documents in two significant ways: first, they are standardized to the HyTime ISO standard to ease portability and market growth; second, they may be expanded by users to accommodate data from online data sources, to provide up-to-date information tailored to the end-user. To do this, MIPS provides facilities (1) for application builders to incorporate queries to local and remote data sources in their application hyperdocuments, and (2) for end-users to issue new queries, the data retrieved in answer to which is used to expand the hyperdocuments. This expansion mechanism requires knowledge of screen, graphic, and other forms of design, currently only being developed in the hypermedia community. The query mechanism itself is driven by queries which are constrained to use the terms which are available in local or remote databases to index data.

[Figure 3 near here]

The public's expectations of the production standards of media products delivered by screen and loudspeaker is high. For MIPS to achieve high production standards for dynamically created presentation mechanisms including video, for example, will require the acquisition and use of knowledge from experts in video editing. Currently, videos have to be retrieved by MIPS in their entirety. In order to be able to select those sections which are best suited to the user's needs, we need a system of tagging by content, with sections of video being accessible by reference to the tags. For example, a surgeon who wishes to compare techniques for the irrigation of a certain part of the body, say, will not wish to examine many hours of videos of operations in which such techniques are used; instead s/he will wish to select only the relevant sequences. Also, to best compose sections into longer presentations it would be helpful to have tags which describe the compositional structure of videos, e.g. 'establishing shot', where appropriate.

To meet these high standards for the automatic retrieval and composition of multimedia assets requires content-addressable multimedia databases which do not yet exist. Research efforts in image analysis may provide prototypes of systems for addressing graphics and photographs by content. Techniques recently described[16] constitute a step towards content-based retrieval for video. MIPS itself does not provide an extensive multimedia database, but the IRL query dialect and the supporting KBS functions have been designed to be applicable to systems we imagine will be developed to support these needs. Online retrieval and composition of video

from remote data sources also requires digital network transfer rates considerably higher than are currently available, in order to make retrieval time bearable for end-users. The present system can be realistically used for the retrieval of relational tables or textual document information over networks, and for the retrieval of other media from local data sources to enable us to explore the requirements of integrating video; however, for this latter function to reach its expected utility, we must await improvements in telecommunications. Although the practical use of MIPS for remote data access of multimedia is not yet commercially viable, for these reasons, it provides a useful step towards this long-term goal by using international standards, linking hypermedia documents to external data, and supporting the tailoring of presentations to the end-user.

Acknowledgement

The work reported in this paper was partly funded by the CEC through Esprit III project 6542, Multimedia Information Presentation System (MIPS). The participating organizations in MIPS are Longman Cartermill (UK), RAL (UK), STI (Spain), SEMA Group Belgium, Heriot-Watt University (UK), Trinity College Dublin (Ireland), DTI (Denmark).

References

1 ISO/IEC JTC1/SC18/WG8, Information Technology — Hypermedia/Time-based Structuring Language (HyTime). ISO/IEC DIS 10744.1.1 (1992)

2 Newcomb, S.R., Kipp, N.A., and Newcomb, V.T. 'The HyTime hypermedia/time-based document structuring language' Communications of the ACM, Vol 34 (1991) No 11, pp 67–83

3 Behrendt, W., Goble, C.A., Hutchinson, E., Jeffery K.G., Kalmus, J., Macnee, C.A., and Wilson, M.D. 'Using an intelligent agent to mediate multibase information access' in Proceedings of the Special Interst Group on Cooperating Knowledge Based Systems (ed. S.M. Deen), DAKE Centre, University of Keele (1994), pp 27–43

4 Jeffery, K.G., Behrendt, W., Macnee, C.A., Wilson, M.D., Kalmus, J.R., and Hutchinson, E.K. 'A model for heterogeneous distributed database systems' in Proceedings of BNCOD 94, Lecture Notes in Computer Science series, Springer Verlag (1994)

5 Cotton, R. and Oliver, R. Understanding Hypermedia, Phaidon Press, London (1993)

6 Rosenberg, J., Cruz, G., and Judd, T. 'Presenting multimedia documents over a digital network' Computer Communications Vol 15 No 6 (July/August 1992)

7 Hutchison, D. 'Distributed multimedia systems' (editorial) The Computer Journal Vol 36 (1993) No 1

8 Davis, H., Hall, W., Pickering, A., and Wilkins, R. 'Microcosm: An open hypermedia system' Proceedings of INTERCHI 93 Conference on Human Factors in Computing Systems, ACM Press, New York (1993)

9 Shneiderman, B., Kreitzberg, C., and Berk, E. 'Editing to structure a reader's experience' in Hypertext/Hypermedia Handbook, E. Berk and J. Devlin (eds), Intertext Publications, McGraw-Hill, New York (1991)

10 Chappel, H., Wilson, M., and Cahour, B. 'Engineering user models to enhance multimodal dialogue' in J.A. Larson and C.A. Unger (eds) Engineering for Human–Computer Interaction, Elsevier Science Publishers BV (North-Holland), Amsterdam (1992), pp 297–315.

11 Bruffaerts, A. (ed.), The definition of the MIPS system, Volume 1: The detailed architectural design of the MIPS system. Deliverable 2.2.2, Esprit III Project 6542 (1993)

12 Van Vyve, J.-M. (ed.), The definition of the MIPS system, Volume 3: The detailed functional specification of the HyTime modules. Deliverable 2.1.2, Esprit III Project 6542 (1993)

13 Buchanan, C. and Zellweger, P.T. 'Automatically generating consistent schedules for mul-

timedia documents' Multimedia Systems (1993) No 1 pp 55–67

14 Chappel, H. and Wilson, M.D. 'Knowledge-based design of graphical responses' in Proceedings of the ACM International Workshop on Intelligent User Interfaces, ACM Press, New York (1993), pp 29–36.

15 Austin, W.J., Hutchinson, E.K., Kalmus, J.R., MacKinnon, L.M., Jeffery, K.G., Marwick, D.H., Williams, M.H., and Wilson, M.D. 'Processing travel queries in a multimedia information system' in Proceedings of ENTER 94: International Conference on Information in Tourism, Innsbruck (1994)

16 Burrill, V., Kirste, T., and Weiss, J. 'Time varying sensitive regions in dynamic multimedia objects: a pragmatic approach to content based retrieval of video'. to appear in Information and Software Technology (1994)

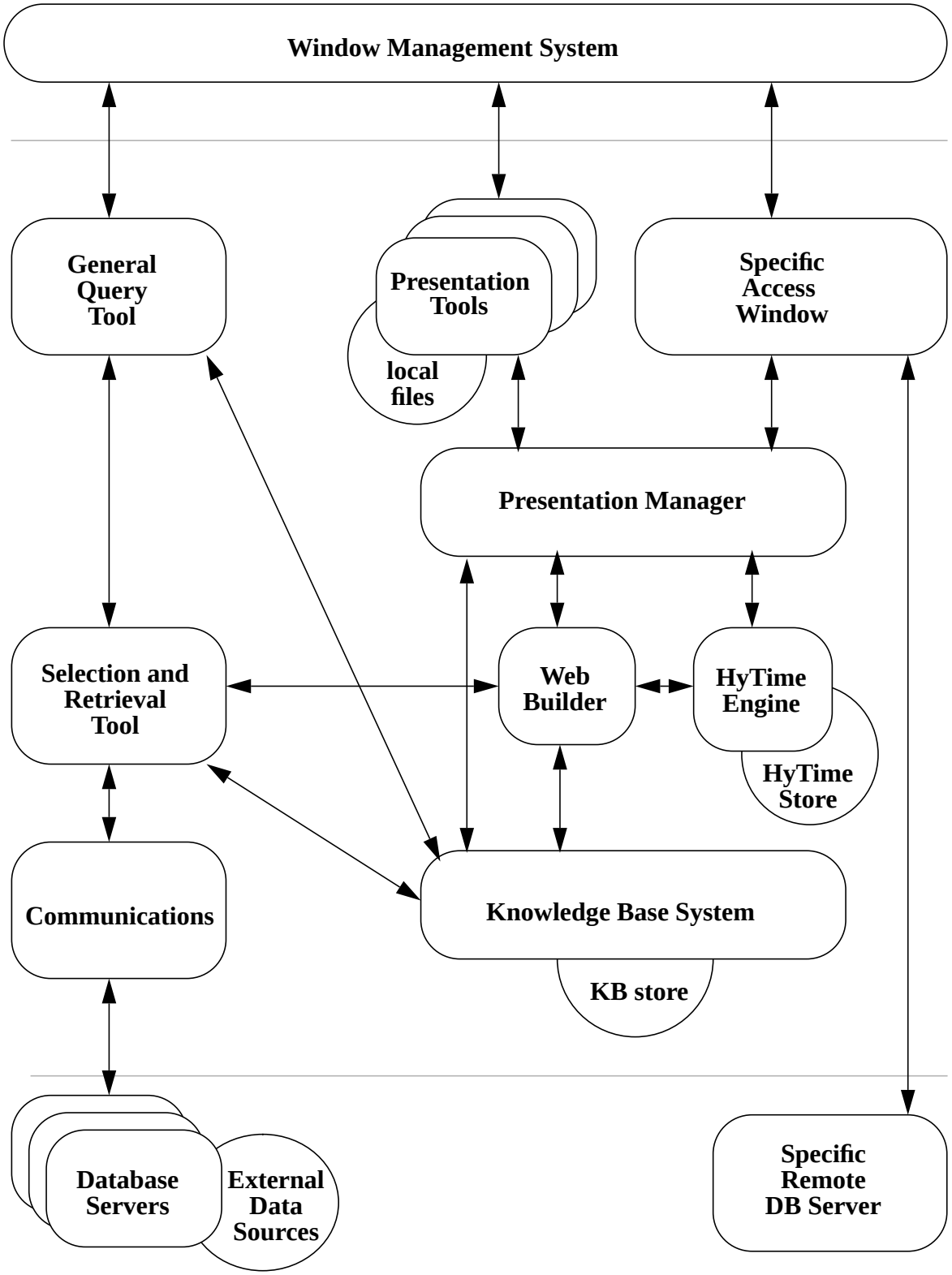

Figure 1: The overall architecture of MIPS.[11]

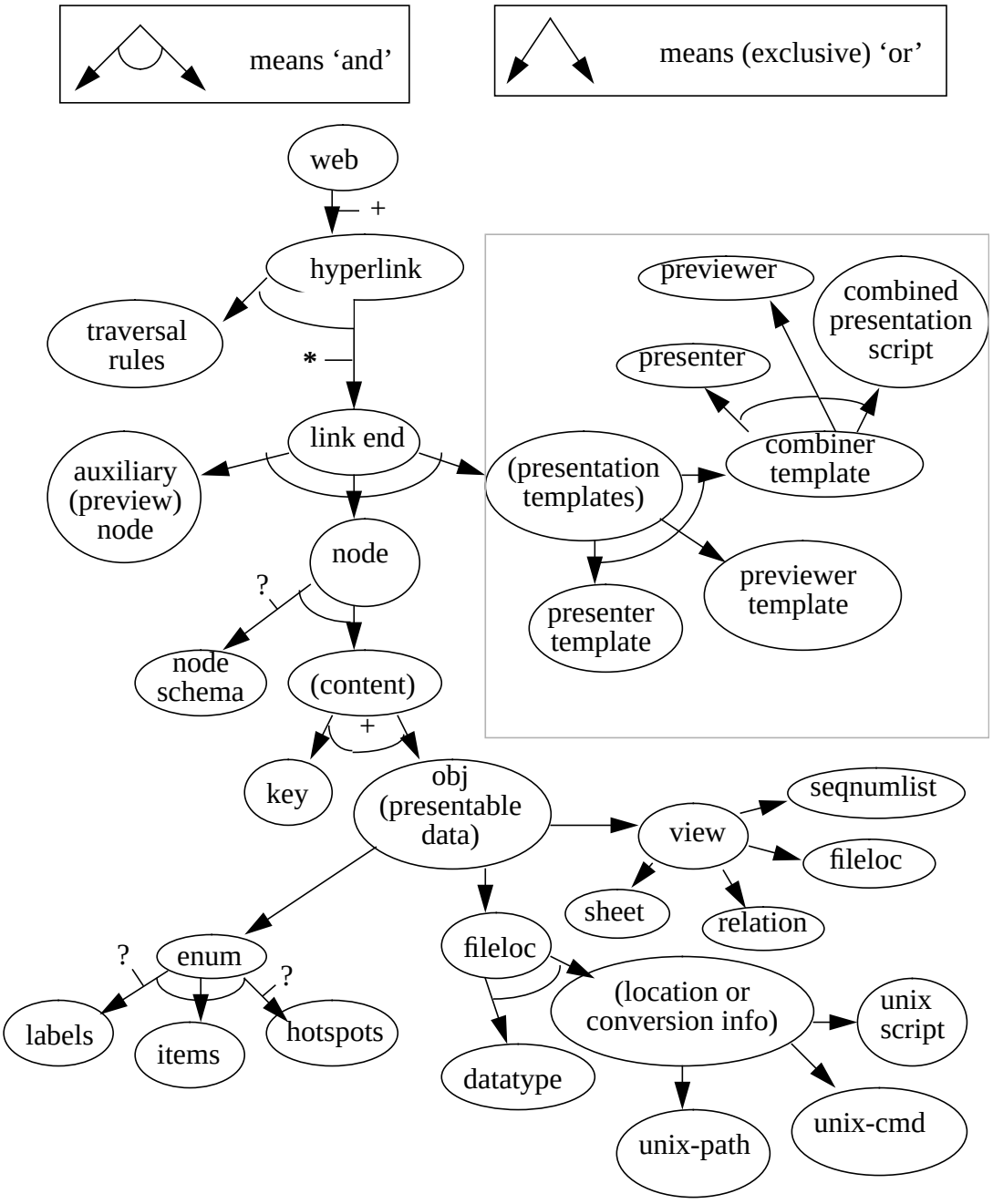

Figure 2: Schematic description of major components of the syntactic structure of the HyTime web. Arrows mean 'refers to or contains' (in each case, either directly or via templates). * means zero or more; + means one or more; ? means zero or one; otherwise, exactly one. The presentation templates in the broken-line box specify the presentation mechanism for the node to which a link end refers.

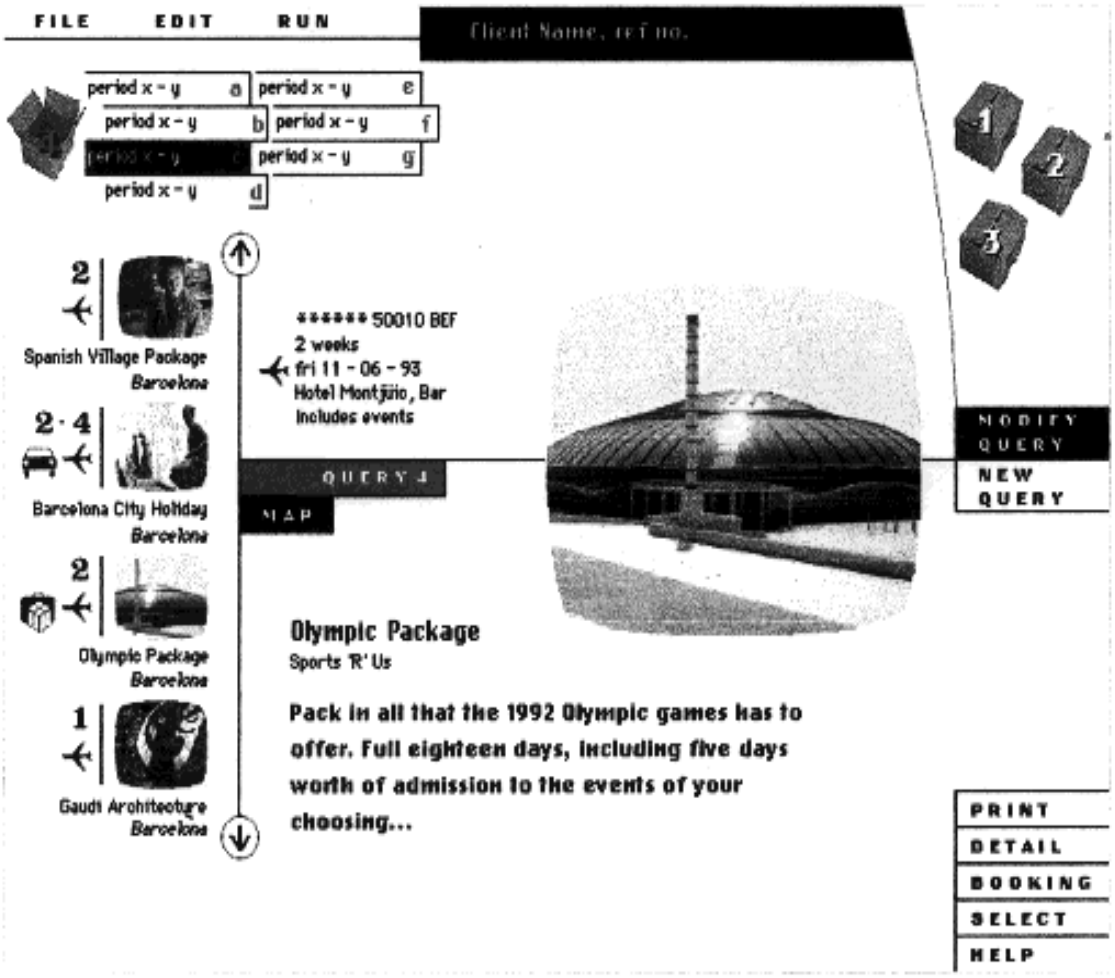

Figure 3: Typical MIPS GUI design.