ISTAR - AN INTEGRATED PROJECT SUPPORT ENVIRONMENT

Mark Dowson
Imperial Software Technology
60 Albert Court Prince Consort Road
London SW7 2BH ENGLAND

1. INTRODUCTION

The need for comprehensive support for the system development process has been recognised for some time. But while there is now a wide variety of tools to aid various development activities, Integrated Project Support Environments (IPSEs) providing comprehensive support for every aspect of software and system production are in short supply.

Part of the reason for this is that most attempts to build support environments have been bottom up. Starting with language specific tools (compilers, linkers, loaders) they have added a superstructure of programming language oriented tools (editors, command interpreters) and run into difficulties providing an adequate database and integrating more general project support tools.

Imperial Software Technology (IST) has adopted the opposite approach. Over the last two years IST has been developing ISTAR, an integrated, language independent, *project* support environment. The ISTAR design process started with a definition of the overall requirements for software project support and the associated database needs. It has led to the development of a comprehensive environment that supports every aspect of software production throughout the life cycle, encompassing project management, data and configuration management and technical development.

A key design objective for ISTAR was to provide the ability to smoothly integrate sets of 'foreign' tools as workbenches that exploit ISTAR's user interface and data management facilities. This ability has been exploited to include workbenches for languages such as C and Pascal, using existing compilers and other language oriented tools.

This approach has now been extended to Ada. The ISTAR Ada workbench includes a validated compiler, and an (initially small) selection of Ada oriented tools. The Ada workbench can be used with the other ISTAR tools to provide comprehensive support for Ada system development projects. ISTAR is thus one of the closest approaches to date to a full APSE.

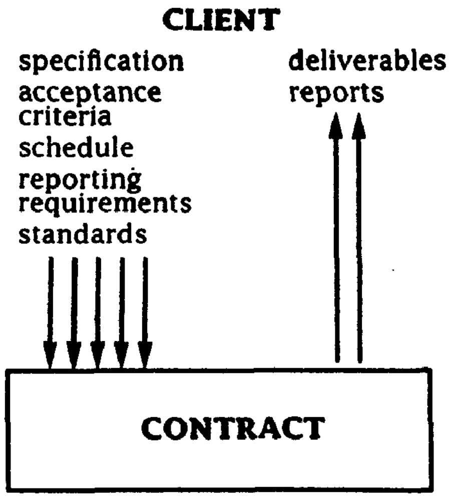
Other ISTAR features include a high degree of portability, and the ability to support distributed projects where development is conducted on a network of host machines.

ISTAR is currently commercially available on a variety of machines running the Unix operating system. Implementations for other operating systems are planned for the near future.

2. THE ISTAR APPROACH

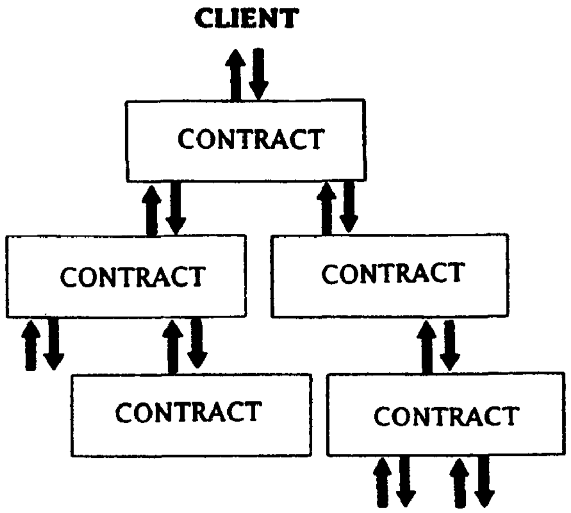
ISTAR is organised to support a powerful but extremely general approach to software and system development, the *contractual approach.*

This approach is based on the recognition that every activity in the software process has the character of a *contract*. That is, an activity is conducted by a 'contractor' eg a programmer or team of programmers, for a 'client' eg a manager. Each activity must have precisely specified deliverables and well defined acceptance criteria for them. In addition, the client may impose other 'contractual' conditions, such as schedules, reporting requirements and technical or management standards that must be followed by the contractor.



Where the size or complexity of a contract warrants, the contractor is free to issue 'subcontracts' to help fulfil the original contract; the subcontractors may themselves issue 'sub-subcontracts' and so on.

The collection of tasks that compose a complete software project forms a *contract hierarchy*. At the root of this hierarchy is the contract for the project as a whole. The leaves of the hierarchy are the self-contained contracts which are completed without letting subcontracts. The intermediate nodes of the hierarchy are subcontracts that themselves let subcontracts. Eventually all the subcontractors will complete their assigned tasks, allowing completion of the original contract.



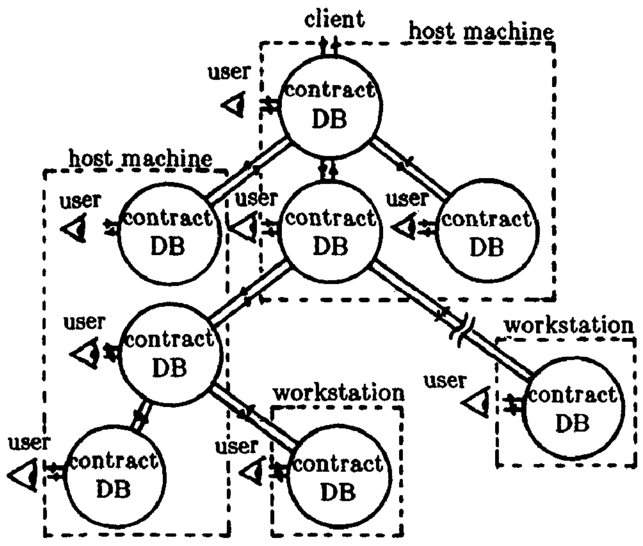
This view does not impose a particular project organisation, or commit the project to a particular model of software development. For example, separate subcontracts could be let for 'specification', 'design', 'coding', 'testing', etc; alternatively, different subcontracts could each be for the complete development of a specific *part* of the system. However, the approach does place some constraints on the ways in which projects can be conducted; in particular it forbids project organisations in which tasks are ill-defined and no one has specific responsibility for executing them.

Explicit adoption of the contractual approach allows a substantial simplification of the infrastructure needed for software project support. Each task can proceed autonomously, recording all relevant information in a strictly local 'contract database'. In addition, formal channels of communication between tasks are well defined, and correspond to the organisation of the particular project.

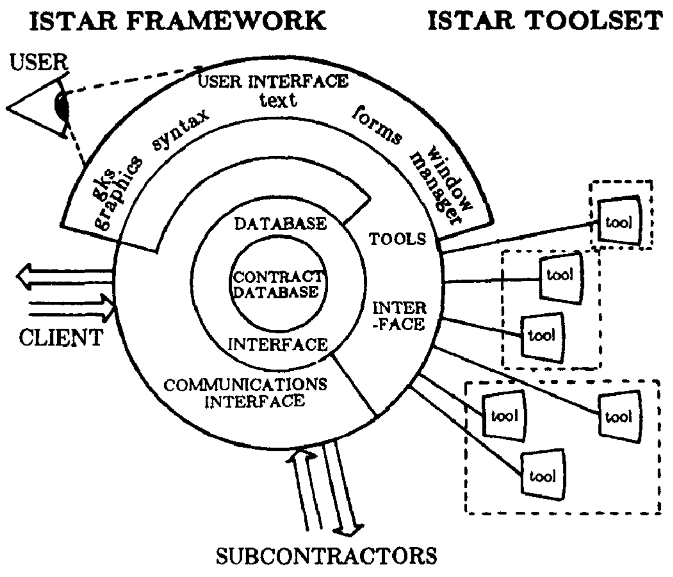
3. THE ISTAR FRAMEWORK

ISTAR consists of two main parts; a framework and a set of tools. The ISTAR framework directly supports the contractual approach described in the previous section by maintaining an independent contract database for each individual contract within a project. When a new project is initiated, a database is created for the root contract of that project, and as new subcontracts are let so new databases are created for those contracts. Thus a direct one-to-one correspondence is maintained through the course of the project between a project's contract hierarchy (organisation) and the hierarchy of contract databases.

True distributed working, either locally or multi-site, is possible by spreading the hierarchy of contract databases over a network of host machines. Work on each contract can proceed autonomously, using the full resources of ISTAR. The majority of tasks, such as contract planning and coordination, and all technical development tasks, are internal to a single contract. The tools that support these tasks operate solely within the confines of the current contract database.



The remaining tasks involve interaction between contracts, in order to initiate a subcontract by assigning it to a particular user at a particular host and creating a new contract database, monitoring its progress, collecting the subcontract deliverable, and so on. The tools that support these tasks still operate primarily on the current contract, but they also use communication facilities to transfer data to and from other contract databases in the hierarchy. These operations are relatively infrequent; thus the communication bandwidth between contract databases does not have to be high and communication can be via local or wide area network or even by physical transfer of media. The resources available to work on each individual contract have four main components: the database system which includes the contract database itself and a database interface; a comprehensive user interface system; a communication system to transfer information to and from other contract databases in the hierarchy; and a set of tools.



A friendly and consistent interface is important if an environment is to be used effectively. The basis of the ISTAR user interface is an 'office automation' quality multi-window text editor which also functions as a forms editor (with local validation of entered information) and as a general syntax directed editor for structured notations. The ISTAR user employs this editor for *all* interactions with the ISTAR framework and ISTAR tools. Windowing is handled by a general purpose, VT100 compatible, window management system which supports overlaid windows, pop-up menus and so on. Users are thus employing the same interface for all activities whether, for example, they are entering a free text description, writing a program, typing a command or filling in a compiler option selection form.

The user interface also provides a small set of standard ISTAR 'function keys' which invoke tool independent operations such as movement between screen windows, display of help information and so on. Display of charts, block diagrams etc is supported (on vector or bit-mapped terminals) by a GKS based graphical presentation system.

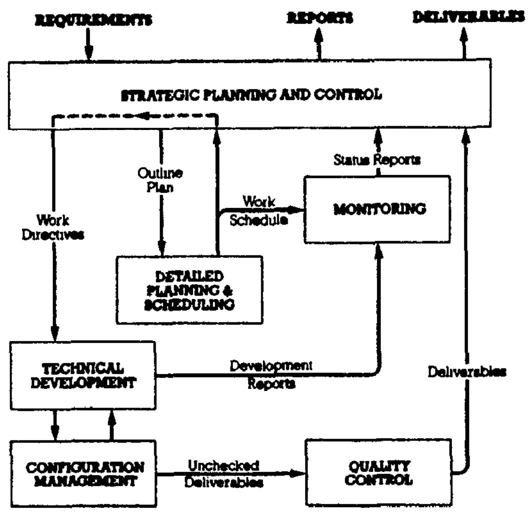
4 ISTAR TOOLS

4.1 INTRODUCTION

While the ISTAR framework does not, itself, constrain the choice of tools that may be used to execute projects, an unstructured collection of tools would not provide coherent support for the system development process. The initial ISTAR tool set - and in particular the project, resource and data management tools - has been designed to supply this coherent support.

Every task in a project requires some degree of planning, scheduling and resourcing. Task execution may produce 'deliverables' that need to be placed under configuration control and checked for quality. Task progress needs to be monitored, and reports on progress may be required. The figure below shows the relationship between these activities in the context of a single ISTAR contract. It should be stressed that ISTAR does not *enforce* the sequence of activities shown, but simply provides support for them if required. The only constraint is that any data exported to other contracts must pass through the contract database and comes under ISTAR configuration control.

DEVELOPMENT ACTIVITIES



ISTAR tools are grouped into *workbenches* - coordinated sets of related tools. Project and Resource Management 'workbenches are available to plan and schedule the work, to assign it to project personnel and to monitor progress; a variety of Technical Development workbenches support the development process from requirement capture through to final implementation; and a comprehensive set of configuration management and quality control tools provide a secure environment for building high quality, maintainable, systems.

An ISTAR user invokes a selected workbench on a particular contract. Thereafter, and until the end of the session, all the tools that are part of that workbench are available to operate on the contents of the database. A few tools, such as electronic mail, are always available, and can be regarded as part of every workbench.

Workbenches are flexible and extendible. For example, while the estimating tool in the current version of the Project Management workbench is based on COCOMO, there is nothing to prevent the substitution (or addition) of another estimating tool based on a different method. This capability allows ISTAR as a whole to evolve either to match changing needs or as new methods and the tools to support them are developed. It also allows ISTAR to be tailored to the requirements of a particular organisation, perhaps incorporating familiar tools, without losing its essential coherence. Direct support for this evolutionary development is provided by the ISTAR tool building tools.

4.2 BUILDING ISTAR TOOLS

ISTAR employs both specially developed tools and existing 'third party' tools that are compatible with the underlying operating system. Third party tools present two potential integration problems. First, these tools will not have been designed to exploit and maintain contract databases, but rather will operate on some collection of files. Second, the tools will not reflect ISTAR's user interface conventions. In order to address these problems, each existing tool is packaged in an 'envelope' a skeletal ISTAR tool that invokes the third party tool, and it is this combination of existing tool plus envelope that is installed as a new ISTAR tool. In such packaged tools, it is the envelope that interacts with the user via the framework user interface system and accesses the contract database as necessary.

Development of new tools and integration of third party tools are supported by a powerful Tool Implementors kit, itself part of the initial ISTAR tool set. The kit includes a set of ISTAR facility libraries for database access, user interface construction and so on, and a special purpose script language for integrating tools into workbenches. A GKS based graphics presentation tool supports the provision of graphical display interfaces for new tools, a flexible report generation language is available for the construction of tailored report generators and an editor syntax compiler and checker allows the addition of syntax capability for new notations to the ISTAR editor.

An important class of technical development tools provide support for structured methods such as CORE or SADT. Such methods build a *specification* according to a *data model* using a generic set of steps which include **analysis** - checking conformance to the model; **prompting** - for missing information; **checking** - against 'advisory' rules; and **reporting** - intermediate and final results. The ISTAR Structured Method Support kit is a kit of generic tools for building support for such structured methods. The kit can be instantiated with a specification of a particular method. To develop support for a new method, the kit is first instantiated with a *method specification method*; then the derived method specification is used to instantiate the kit for the method to be supported.

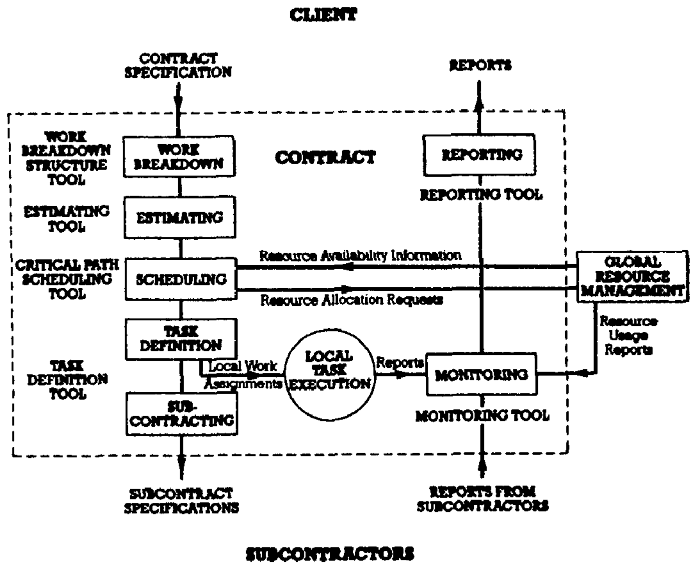
4.3 INITIAL ISTAR TOOLS

ISTAR tools fall into six broad classes: management tools; technical development tools; data management tools; office automation tools; tool building tools; and system administration tools. As described above, in each class the tools are grouped into workbenches. For example, the Project Management workbench includes tools for planning, estimating, monitoring and contract management, all of which can be used in the same session on the same contract.

Project and Resource Management

Management tools are central to ISTAR. They support the contractual approach to development, and make possible the coherent organisation of a project so that the technical development tools can be deployed effectively. The figure below shows the relationship of the various project management activities in the context of a single ISTAR contract.

PROJECT MANAGEMENT



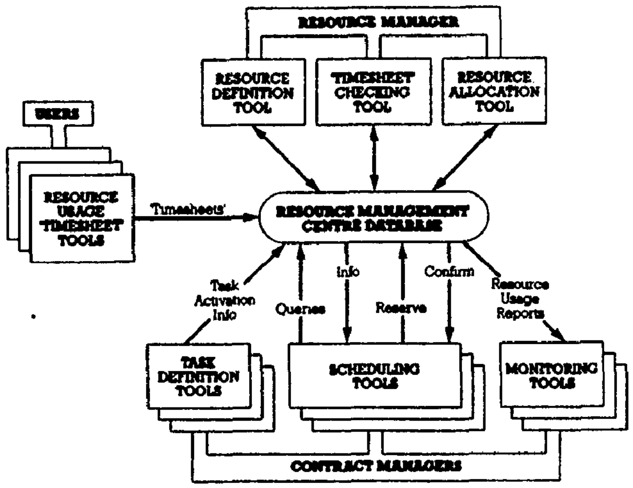
Work Breakdown Structuring is a fundamental activity which divides the assigned contract task into work packages which can either be performed as part of the current contract or subcontracted further. The ISTAR Work Breakdown Structuring tool provides three coordinated viewpoints to support the breakdown of tasks into subtasks: the *activity view* of the component activities of each task and their dependencies; the *product view* identifying the products of each activity and defining a product hierarchy; and the *resource view* associating resource requirements with activities and allowing resource estimates to be checked against constraints.

The Scheduling tool is a critical path scheduler that can work in both batch and interactive modes. By communicating with the ISTAR Resource Management tools it makes allocations of suitable resources to activities following an automatically generated resource limited critical path schedule. Manual intervention with the scheduling process allows the schedule to be modified to meet additional constraints and permits 'what if' exploration of schedule changes.

Once again, ISTAR does not enforce the use of these tools or require them to be used in the sequence shown (with the exception of the Task Definition tool which is required for the assignment of subcontracts). For example, the Estimating tool, a full implementation of COCOMO, need not be used at all, and in any case is likely only to be used at the higher levels of a contract hierarchy. Similarly, users are free to ignore the progress information collated by the Monitoring tool, and to transmit intuition based free text reports to the contract client if they so wish (and if their manager permits).

Resource management is a complex activity, and ISTAR provides a powerful set of tools to support it. ISTAR resource management is based on the establishment of Resource Management Centres each with their own database of resources. Resource Management Centre databases are normal ISTAR contract databases, and the resource managers are normal ISTAR users who have been assigned a contract to manage a set of resources. Resource Management centres may be used to control the allocation of resources within a single project or across projects on a company- or division-wide basis, and may hold mixed resources - staff, space, computing resources - or be dedicated to a single resource type. The structure of Resource Management Centres, and the tools used, are shown below.

RESOURCE MANAGEMENT

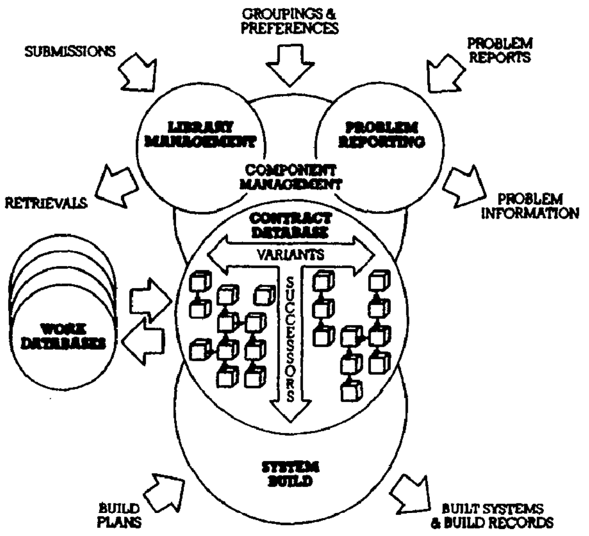


Resource management starts with the definition of the set of resources to be held at the centre, using the Resource Definition Tool. The Resource Management Centre database is interrogated by contract managers who then make resource reservations which can be arbitrated by the resource manager using the Resource Allocation tool. Activated tasks can be booked to by users and the bookings checked. Finally, resource usage reports are returned to the appropriate contract for integration into contract progress reports.

Data and Configuration Management

Data management is integral to the structure of ISTAR. During the course of executing a contract, data items are created and modified by ISTAR tools in *work databases* and exported to the contract database where they come under strict ISTAR configuration control. Contract databases have a built in versioning mechanism, supporting both *successors* which supersede the previous version and *variants* which undergo parallel development. In addition, the database can hold data item attributes (such as 'preferred version') and user defined relationships between items (such as 'uses the same widget as'). Data items that have any dependencies are 'frozen' and cannot be deleted, and ISTAR keeps track of any data items that are exported to other contract databases. A number of tools are available to control and manage data. Their relationships are shown below.

DATA MANAGEMENT



The component management system allows the user to inspect the content of the contract database, navigating trees of variants and successors; to copy data items for the creation of new versions; to delete redundant (non-frozen) data items; and to define data item attributes and relationships. It is also used to provide basic services for the library management and problem reporting tools.

Libraries in ISTAR are contract databases containing shared configuration items. Each has a *librarian* - a user who deploys the ISTAR Library Management tool to control submission and distribution of library items, to commission work to correct errors in items and supplement the content of the library.

The Problem Reporting tool allows the generation of problem reports about particular data items, submission of the reports to e.g. the contract database where the item is held, and dissemination of reports to other users of the item. Initially, the originator of a problem report is its *coordinator* and has responsibility for progressing it. But the role of coordinator can be passed to another user e.g. the creator of the faulty item or the librarian of the library where the item is held.

The Build workbench provides a set of tools for building executable programs for release. A command list specifies the names of the needed transfer items containing e.g. source code, and what actions on them e.g. compilation, are required to complete the build. In addition, a bindings list indicates which item versions are to be used. The component transfer items are imported into the build work database from the local contract database and from remote libraries. When the build is complete, the built system and a build record are exported to the contract database where they are subject to configuration control.

Technical Development

Technical development tools support the technical activities of a contract, including production and verification of the contract deliverables. Tools supporting the programming process, for example compilers, debuggers and test tools are available from a variety of sources. The ISTAR technical development tool set includes a number of workbenches composed largely of integrated collections of such existing tools, packaged where necessary to provide a common interface style for the workbench and integrated access to the contract database. In general, these workbenches are language specific. For example, the C workbench includes a syntax directed editor for the C language, a C compiler, linker and dynamic debugger, and a set of C oriented test tools. Initially, language oriented workbenches are being provided for C, Pascal, Ada and CHILL; they are mainly composed of existing tools, but incorporate specially developed tools where necessary.

In the longer term, IST intends to release a number of Ada workbenches. These will differ both in the Ada compiler used and in the mix of additional tools. For example, an Ada 'cross-compiler' workbench might include a target machine emulator and/or target oriented remote debugging tools. The initial Ada workbench includes the (validated) Alsys Ada compiler, associated linker, debugger and cross reference lister and the ISTAR syntax directed editor. The compiler is integrated to the extent that it uses normal ISTAR contract databases as package libraries, allowing the full range of ISTAR configuration and data management tools to be used for Ada program development.

The area of specification and design methods is not nearly as well served by existing tools, and ISTAR workbenches for them will mainly consist of specially developed tools. Initially, workbenches have been developed to support the CORE Controlled Requirements capture and Expression method (built using the Structured Method Support kit); VDM, the formal Vienna Development Method for system specification and design; and SDL, the CCITT recommended Specification and Description Language for concurrent real time systems.

Quality Management

ISTAR does not enforce a particular quality control policy, but provides support for implementing the chosen policy using the Quality Assurance (QA) check list mechanism. QA checklists may be passed down to subcontractors as part of the contract specifications, used by the client of a contract to check the quality of received deliverables, or passed to a separate 'quality control' contract. QA checklists can contain references to other QA checklists (which might be held in a QA checklist library), allowing a hierarchy of QA checks to be applied.

Office Automation and System Administration

ISTAR is intended to support all the activities involved in conducting a software project, many of which are clerical and not specifically related to software or system development. Thus, ISTAR includes office automation quality tools for activities such as document preparation and production, electronic mail and so on, all accessible through the standard ISTAR user interface.

In addition, ISTAR includes a comprehensive set of system administration tools supporting such functions as ISTAR network configuration, adding and removing users, archiving and database recovery. The presence of these tools, together with the useability of the other ISTAR tools has led to the (as yet unproven) suggestion that the learning time for ISTAR is negative - compared with raw Unix.

5 SUMMARY

ISTAR is a full, commercially available, integrated project support environment. It includes a comprehensive and extendible set of tools covering every aspect of the software and system development process.