1992 ISS XIV International Switching Symposium Yokohama, Japan October 25-30, 1992

# Spartan: A Data Design System for a Real-time Memory-resident Database

Ronald J. De Lange, Eisuke Muroga, Yao-Nan Lien, and Michael J. McPheters

AT&T Bell Laboratories, USA

## ABSTRACT

Spartan is a data design automation system for the 5ESS® Switch real-time memory-resident database. The 5ESS Switch architecture and application software requires the database to be designed with speed, space efficiency, and correctness. The database design process is comprised of five unique phases, each with its own requirements: Investigation, Design, Engineering, Approval, and Verification. The design of Spartan incorporates distinctions between the process phases, yet provides a single environment to support all the phases. The Spartan System has been implemented and adopted, resulting in an improvement of 5ESS Switch database quality, and a reduction of its design time interval and cost.

## 1. INTRODUCTION

The Spartan System is an integrated system to assist and support the design of the database of the 5ESS Switch software architecture. The need for the Spartan System arose because the data design task is very complex, yet critical to the performance of the 5ESS Switch database.

The software to process telephone calls on a 5ESS Switch is highly dependent on a real-time, memory-resident database [Barc82, Dela85, Kukl87]. This database allows each switch to be customized for a particular telephone office. The performance and the reliability of the switch is highly dependent on the correctness, speed, and efficiency of this database.

The 5ESS Switch database uses the relational database model. In a relational database, data is stored as sets of *relations*. Each relation contains several fields of data called *attributes*. The set of allowed values for each attribute is its *domain*. For each relation, there may be one or more uniquely-valued entries, or *tuples*. There can also be *views* defined on the database, which present sets of cross-referenced attribute values from one or more relations. Any semantic rule covering the legal values in the database can be defined by *population rules*, including the cross-references between office features, views, relations, attributes, and domains.

The unique aspects of the 5ESS Switch database necessitate the creation of a specialized data design support system:

1. The database is extremely large and complex, comprised of about 1000 relations and views, as well as hundreds of attribute domains. In addition, there are thousands of population rules used to enforce the integrity of the relations in the database.
2. Stringent memory space and speed requirements to support a real time system (tuple access time is in the μs range) makes access and storage tuning necessary.
3. The database is distributed among the multi-processor architecture of the 5ESS Switch.
4. The database is constantly evolving, and all structure changes must be coordinated with many other changes.

® Registered trademark of AT&T.



The data design and development process has several phases (detailed in section 2): Investigation, Design, Engineering, Approval, and Verification. Before changing the database structure or population, a data designer needs to perform an initial investigation of the existing designs, since comprehending a database of 1000 relations is a non-trivial task. Furthermore, with several hundred engineers simultaneously developing 5ESS Switch software, it is necessary for all data designs to be coordinated, ensuring that there are no conflicts or overall inefficiencies. This process was previously done manually, aided by a collection of tools supporting isolated tasks. Thus, this process was time-consuming and required a verification process that was highly dependent on human experts to ensure its quality.

The objective of the Spartan System is to provide an integrated design environment that encompasses all data design and development phases to provide guidance, assistance, and automation for the data designer, resulting in efficient and error-free changes to the 5ESS Switch database. The Spartan System provides a menu-driven interactive environment with a similar "look-and-feel" user interface to support all five phases. It also provides a consistent working environment and a mechanism to store and share information between the five phases. This coherence ensures that data designers can work under the same environment as other data designers, whether working on the same project or on different projects. Spartan also offers a wide selection of previously-unavailable tools that support data design and testing. By significantly enhancing the quality of data designs, the Spartan System has become an integral part of the data design process. The Spartan System will be described in more detail in Section 3.

## 2. INTRODUCTION TO 5ESS SWITCH DATA DESIGN METHODOLOGY

This section describes each of the phases of the current 5ESS Switch data design process. The user of this methodology (the data designer) is generally a 5ESS Switch software application developer whose requirements include data in the 5ESS Switch database. The process is divided into five phases that were introduced in the previous section: (1) Investigation, (2) Design, (3) Engineering, (4) Approval, and (5) Verification.

### 2.1 Investigation Phase

Each data designer must identify and understand the data in the 5ESS Switch database that he/she must use. The application

software requirements and existing call data flows are used to compare the requirements with the current data designs.

The data designer also identifies any other data that potentially may be affected. To integrate new data designs with all existing data designs for syntactic and semantic consistency, the data designer investigates the potential impact on the existing database by the new data design. Existing data designs are often used as templates for new designs.

### 2.2 Design Phase

With the design requirements set and the existing data designs fully investigated, the data designer specifies the logical structures of the necessary data. This may result in the creation of a new view based on existing relations, or the modification of an existing view or relation, or the creation of a new relation. In this phase, the data designer defines the necessary attributes and domains in the relation definition language called RDL [Brai90]. The physical and logical storage locations for the data (called relation partitioning and distribution) are also defined in this phase [Barc82].

During this phase, switch application software such as the Recent Change and Verify (RCV) system is examined for potential design changes or impacts. The RCV system is used to administer the on-line database on an 5ESS Switch [Phil84].

### 2.3 Engineering Phase

An unusual aspect of the 5ESS Switch database is that the access method for each relation must be optimized for speed and space efficiencies. This important per-relation performance tuning can improve the database performance, and thus the performance of the 5ESS Switch as a whole.

Tuning a real-time memory-resident database system for efficiency differs from that of a traditional disk-based database system in many ways, among those being:

1. Minimizing memory usage in a memory-resident database is essential due to limited memory space.
2. Specialized access methods are required on a per-relation basis.

Some typical engineering issues of data design include access type selection, attribute packing into words (to fully exploit the physical memory byte structure), relation partitioning and distribution design, and specification of optimization parameters. Parameters used include the maximum allowable number of tuples, the size of the tuples, the type of keys, and the anticipated access procedure (e.g. linear, random).

### 2.4 Approval Phase

Since the 5ESS Switch software is driven by the database, any change to the database potentially impacts many software modules that are owned by different organizations. Coordination among the data designers and the impacted organizations is essential to maintain consistency between the database and all associated software modules. To help identify all affected software modules and organizations, each database change is categorized according to the following factors:

1. The *degree of impact*, ranging from minor to critical
2. The *type of data being changed*, such as relation name or data associated with a 5ESS Switch feature
3. The *subsystems that may be affected*

Thus, each data design goes through a formal approval phase to ensure that each data design is consistent with other designs. Two forms of documentation are used to describe a data design change. The first type of documentation details the Investigation, Design, and Engineering phases of the design, and the analysis tasks associated with each phase. For example, a domain change must be syntactically and semantically correct. Furthermore, changing a domain's size may also cause a relation's memory usage to increase; this must have been modeled and analyzed for memory efficiencies. These analysis outputs are reviewed as part of the approval process.

The second type of documentation is a form of design documentation that has defined sections for each type of data design. Tables are used to indicate all the various areas and specific domains, relation structure definitions, and constraint rule files impacted or associated with the data design.

### 2.5 Verification Phase

Another difficult issue in data design is the maintenance of data integrity. The integrity constraints in the 5ESS Switch, called *population rules*, are defined along with relation definitions and are independent of switch application programs. The population rules are written in a language called *Population Rule Language* (PRL) [DiPi88], which is used to perform large-scale cross-checking of data integrity.

During the software development process, test databases are populated and used by data designers. A verification process using the PRL integrity constraints is used to validate the integrity of the database. Any error found is analyzed and corrected.

## 3. THE SPARTAN SYSTEM

This section describes how each of the five phases of the 5ESS data design process are supported by the Spartan System. The Spartan System was conceived as a system to encapsulate the entire data design process (as outlined in section 2), but it was found that each phase of the process had unique needs. Thus the design of Spartan incorporates distinctions between the process phases, yet provides a single, consistent environment for all the phases.

In section 3.1, a general picture of the software architecture of Spartan is presented. Then, in the ensuing sections (3.2 through 3.6) details of Spartan are described for each of the five process phases introduced in section 2.

### 3.1 General Architecture

The Spartan system covers the entire data design process, but is organized so that each phase of the process is supported by a specialized Spartan software sub-module.

The architecture of the Spartan System is designed to support separate, yet interdependent process phases. The key feature of the Spartan architecture is how various data is stored, accessed, and shared by each of the modules. The data in the private databases are used only within a particular module, whereas the shared databases are used not only as a common repository of data, but as the underlying link between the modules.

The shared databases are:

1. *Change Management System* - a database system that controls all changes to 5ESS Switch software and data-definition source code [Basi85, Tusc87].
2. *Version Repository* - a read-only repository containing snapshots of a complete version of 5ESS Switch source code [Basi85, Tusc87].
3. *Private Source Set* - a set of source files privately held by each data designer.

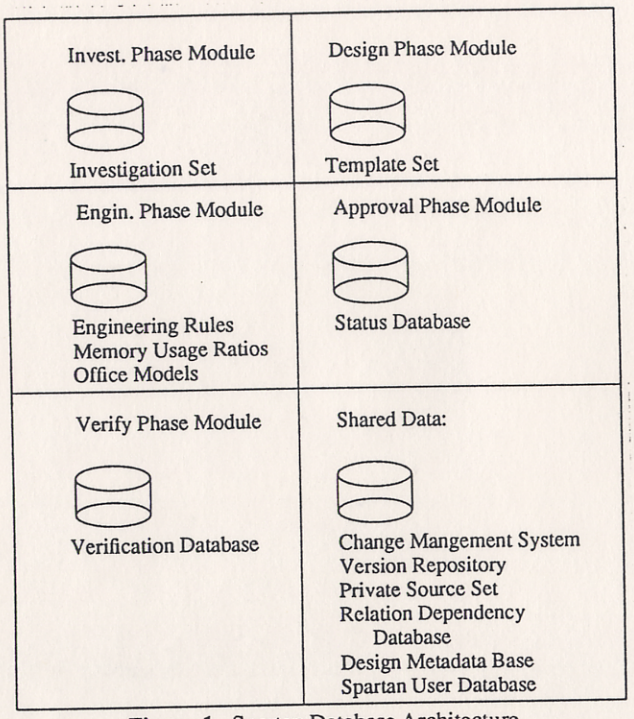


**Figure 1.** Spartan Database Architecture

4. *Design Metadata Base* - a database containing metadata describing the designs contained in the Version Repository.
5. *Relation Dependency Database* - a database containing dependency information between all relations.

How these databases are used by each module is discussed in later sections. Each module also has private databases that are described in more detail in the ensuing subsections of Section 3, but are briefly defined here. The Investigation phase uses a temporary repository of query results in a private database called the *Investigation Set*. The Design phase uses a *Template Set* where design templates are stored. In the Engineering phase, there are three important databases used: the *Engineering Rules* database, the *Memory Usage Ratio* database, and the *Office Models* database. The Approval phase has a database where various status information is kept. The Verification phase also uses a private database called the *Verification Database* to hold information on verification histories.

### 3.2 Investigative Phase

The Spartan System supports the Investigation phase by providing access to information on past data designs. The importance of providing this access to the designs is twofold:

1. The data designer can reuse parts of existing designs.
2. The data designer can see what other parts of the database will be affected by any changes the designer makes.

A database of information regarding all existing relations and domains is maintained. To assist the data designer in browsing through the existing data designs, the Spartan System incorporates a set of investigation tools.

There are three basic tools that allow the data designer access to the *Design Metadata Base*. These include two browsers: one for relations, and one for domains. Each browser can display either relations or domains that meet a set of criteria selected by the data designer. The predefined criteria can be chosen and combined from a menu of criteria, much like a query. The set of relations or domains that result from the query selections is kept in a temporary private database, called the *Investigation Set*.

The third investigation tool is a *relation dependency analyzer* (RDA). A change made to a relation is likely to impact another relation that is dependent on the first relation. Thus, the data designer must find all relations that are dependent upon the relation being modified. It would be a complex and time consuming task to identify all inter-relation dependencies manually. Spartan can construct for the data designer the entire transitive closure of the dependency by abstracting the dependency relationship from population rules. This dependency information is stored in the *Relation Dependency Database*. Users can choose to browse for the dependency interactively through the RDA or to see a set of dependencies as a dependency graph.

### 3.3 Design Phase

The Spartan System provides an interactive development environment to privately develop and analyze the physical and logical designs of new or modified relations. Each new or modified relation in the switch database must be considered in terms of where the data is to reside (which processors), how the data is to be accessed, and how the data is to be stored.

Existing relation or domain physical definitions can also be modified. Spartan provides a set of templates to help data designers quickly construct an initial physical design for new relations or domains. Spartan provides the mechanism to use a copy of a relation or domain definition file from the Version Repository. The data designer is placed in an interactive mode using the Relation Definition Language (RDL) to update the template copy and manage it in the data designer's *Private Source Set*.

After the data designer has created new domains or relations and modified copies of any existing data structures to be changed, the *Private Source Set* is then syntax checked and a private *Design Metadata Base* is constructed or updated to provide a platform to describe the data's logical relationships.

RDL syntax and physical design engineering rules are referenced from the *Engineering Rule Database* stored within the Spartan System. A syntax directed interface/editor is provided that has knowledge of all relations and domains defined in the database via the private *Design Metadata Base*. This interface/editor is used to define logical data constraints and inter-relation dependencies.

Once the basic design has been privately implemented, design documentation can be automatically generated and reviewed by the design team. By providing these capabilities in a private environment, many different design alternatives can be quickly defined and considered.

### 3.4 Engineering Phase

The Spartan System Engineering phase is designed to support data designers in optimizing the physical aspects of their designs for speed and space efficiency.

Spartan provides an interactive ability to forecast memory usage based on the number of anticipated tuples for a new relation. Alternatively, Spartan can estimate memory usage based on simulated or real data provided as input. This can be especially useful when modelling different types of switch configurations. Using simulated or real data as input, Spartan can also characterize the populated relation in terms of data clustering around logical keys.

For modified relations, the number of anticipated tuples and/or the memory utilization rate (MUR) calculated from data collected during the 5ESS Switch's transition from one software release to another is used to forecast memory usage. The *Memory Utilization Ratios* database is used to obtain the MUR data.

Each relation structure definition can be viewed on the data designer's terminal with the bit layouts of each attribute displayed. Attributes can be moved around interactively to find the minimum memory usage. Spartan can also automatically suggest a relation structure layout that minimizes memory usage. By using the private *Design Metadata Base* and *Private Source Set*, the engineering aspects of the design can be trialed to find the optimal physical design.

The data designer can use default values or specify a sub-set of engineering characteristics for a particular relation. Spartan, referencing the *Engineering Rules* database, will suggest the optimal relation access method to be used by the 5ESS Switch Data Base Manager. Spartan can also summarize existing engineering for similar relations using the *Design Metadata Base*, which can be used to model a new relation.

### 3.5 Approval Phase

The Approval process has several components: 1) Place the data design changes into the *Change Management System*. 2) Syntax check and compile all affected files associated with the design after insertion into the *Change Management System*. 3) Construct, schedule, and review the data design in a package format. 4) Automatically generate data design documentation pieces for inclusion in the data design memorandum.

All final descriptions of the data design must reside officially in the *Change Management System*. Until the approval process has been initiated, all data design changes are made privately to trial-and-error various design alternatives with no impact to other developers. The Spartan System gives the developer the ability to manage his/her private changes within their *Private Source Set* into logical groupings related to when the changes will be formally submitted for approval.

Spartan then offers the developer the ability to syntax check and compile all the affected files associated with the data design. Spartan uses an extracted version of the official data design from the *Change Management System*. This process step assures the designer that the changes migrated from the *Private Source Set* to the official *Change Mangement System* have transitioned accurately, and that any other developer's changes are known. Additional outputs of the compilation process are memory utilization and efficiency analysis reports.

The official changes for the data design and the resultant compilation outputs are used by Spartan as inputs to construct a data design package. Spartan also provides an interactive interface to provide information and technical details related to the design. Spartan identifies who the appropriate reviewers and module owners are for each area the design affects and notifies them of the impending changes. Additionally, the developer is scheduled for a review with representatives from the impacted areas to formally review and approve the data design package. Spartan automatically manages the approval transition states that each package migrates through and informs the package owner and impacted area representatives when state changes occur. The *Status Database* manages these state transitions as well as information related to who is impacted by the design.

Finally, Spartan automatically provides the textual inputs for inclusion into the data design memorandum associated with each design. All affected relations and/or domains, memory utilization estimates reports, and analyses are generated.

### 3.6 Verification Phase

Under the current 5ESS Switch data development process, the integrity constraints of each relation are defined in a set of population rules using PRL. These population rules can be used for data integrity maintenance as well as for documentation described as follows:

1. All 5ESS Switches and all testing laboratories of the same software release populate their databases, called *Office Dependent Data* (ODDs), according to the same set of relation definitions and population rules [Baue85].
2. After an ODD is populated, an ODD verification process is carried out to check its integrity using a executable set of population rules.

Although many commercial databases use a query-language-based integrity constraint system (e.g. SQL) [Date89, Codd90], the specialized constraint language PRL was created to better handle the needs of the 5ESS Switch database. The constraint language is used to define thousands of rules, making it a very large and complex rule-set. The PRL is compiled into an executable constraint-checking program, allowing a populated database to be checked against these rules.

The ODD Verification Process, which is encapsulated in the Spartan System, is discussed in the rest of this section.

To verify the correctness of an ODD, the Spartan System is invoked to check every tuple of each relation, which may be replicated in multiple processors. The Spartan System will flag an error if an inconsistency is found between the tuple and the corresponding rule. After the check is completed, the errors found must be analyzed to determine their root causes, and then be corrected. There are four possible causes:

1. *population rule error:* the population rule itself is not correct.
2. *data entry error:* the tuple is incorrect due to an error introduced during the data entry process.
3. *data administration tools error:* the tuple is incorrect due to an error introduced during an off-line data modification or creation process.
4. *simulated error:* the error is embedded into the database intentionally for testing purpose.

One of the biggest challenges to the ODD error analysis process is to identify unique new errors from amongst a large number of redundant errors, including simulated and analyzed (but uncorrected) errors. The Spartan System maintains a *Verification Database* containing information of all errors, including an analysis history. The Verification Database contains a log of all previously found errors (whether corrected or not), correlated across labs and across versions (snapshots in time) of the data in the database. Any errors that have been corrected also have root cause analysis information stored in the Verification Database. By consulting this database, the Spartan System can help easily identify new errors and analyze the root cause(s). The information regarding errors is presented to the data designer in a concise and readable format.

Finally, the Spartan System is able to provide graphical error statistics. This capability is particularly useful for project management and error tracking.

It is important to ensure that a set of population rules is correct before they are applied to an actual 5ESS Switch database. Very useful for debugging population rules are the switch-testing laboratory databases. These testing switches have databases that are very small, well-defined, and very stable. The data verification process begins with these testing databases, not only to verify the data itself, but also to verify that the population rules themselves are correct. As a result, a very accurate set of rules is available to check all ODDs of real switches after this initial testing for a new software release.

## 4. CONCLUSIONS

Before the Spartan System was adopted, the data design and definition process was minimally documented and supported. Each developer had to discover what the processes were and what engineering guidelines were applicable to their design. Data design was complex, tedious, and very time consuming.

By formalizing the process via the Spartan System, the processes to design data are well defined, with known entrance and exit criterion. Providing a common framework and languages specific to data design via *Private Source Sets* facilitates private, quick, interactive definition as well as tools that ensure the quality of the 5ESS Switch database and its data. The resulting automated code generation, verification processes, and common interfaces have significantly reduced the design interval and costs, and have improved the quality of data designs much earlier in the software development process.

There are several improvements planned that will increase the effectiveness of Spartan, and thus the quality of the 5ESS Switch database.

There is a plan to make data designs under Spartan self-documenting. The data design process already has some self-documenting features such as the maintenance of the Design Metadata Base, but further steps will be taken to produce human-readable reports on the data designs created under Spartan.

There are further efforts to fully automate the Engineering phase, so that the data designer need only specify some high-level parameters. Ultimately, these engineering parameters also will be automatically generated for the data designer.

There are efforts underway to incorporate more design knowledge into the Spartan System. The process guidance will be proactive, in that not only will there be process guidance in the form of on-line expert "advice," but the process steps will be enforced.

## 5. ACKNOWLEDGMENTS

The authors would like to acknowledge Shafik Hakim, Leszek Lilien, and Wen-Bin Lo who participated in the inception of Spartan. Many thanks to our reviewers: Bernard Sander, Chris Ramming, Mark O'Connor, Diane Pedersen, and Bharat Desai for their constructive critiquing.

## REFERENCES

[AT&T85] AT&T: "IFS - A Tool to Build Integrated, Interactive Application Software," AT&T Technical Journal vol. 64 in November '85.

[Barc82] D. K. Barclay, E. R. Byrne, and F. K. Ng, "A Real-time Database Management System for No. 5 ESS," Bell System Technical Journal Vol. 61, No. 9, Pt. 2, Nov. 1982.

[Basi85] R. G. Basinger, J. A. Herndon, B. Kaskey, J. A. Lindner, and J. M. Milner, "The 5ESS Switching System: System Development Environment," AT&T Technical Journal, Vol. 64, No. 6, Pt. 2, 1985.

[Baue85] H. A. Bauer, L. M. Croxall, E. A. Davis, "The 5ESS Switching System: System Test, First-Office Application, and Early Field Experience," AT&T Technical Journal, Vol. 64, No. 6, Pt. 2, 1985.

[Baum81] S. M. Bauman, R. J. Carline, J. S. Nowak, and H. Oehring. "No. 5 ESS Software Design," Proceedings of International Switching Symposium, 1981.

[Brai90] R. M. Braida, B. N. Desai, P. Fu, and D. W. Walker, "RDL Reference Manual and Design Guidelines for Operational Views," AT&T Bell Laboratories internal document, 1990.

[Codd90] E. F. Codd, *The Relational Model for Database Management: Version 2*, Addison-Wesley Publishing Company, 1990.

[Date89] C. J. Date, *A Guide to the SQL Standard (Second Edition)*, Addison-Wesley Publishing Company, 1989.

[Davi81] J. H. Davis, J. Janik, Jr., D. Royer, and B. J. Yokelson. "No. 5 ESS - System Architecture," Proceedings of International Switching Symposium, 1981.

[Dela85] J. P. Delatore, R. J. Frank, H. Oehring, and L. C. Stecher, "The 5ESS Switching System: Operational Software," AT&T Technical Journal, Vol. 64, No. 6, Pt. 2, 1985.

[DiPi88] J. E. Di Pirro, "Population Rule Language (PRL) Reference Manual," AT&T Bell Laboratories internal document, 1988.

[Kukl87] Jim A. Kukla and Fred K. Ng, "A Highly Reliable DBMS for the 5ESS Switching System," 1987 IEEE Conference on Computers and Applications, 1987.

[Phil84] Phillip T. Fuhrer and Paul Moscoso, "Data-base administration for the 5ESS Switch: flexible, simple," AT&T Bell Laboratories Record, 1984.

[Tusc87] P. A. Tuscany, "Software Development Environment For Large Switching Projects," Proceedings of International Switching Symposium, 1987.