

23

A FULL-SCALE IMPLEMENTATION OF SOFI

By

Peter P. Yim, Jonathan M. Cheyer and Theodore J. Gordon

I. History of the SOFI system

II. What is the SOFI System Application

III. How to use the SOFI System

IV. Strengths and Weaknesses

V. Use with other Futures Research Methods

VI. Speculations about Future Development of the Software

VII. Appendix

Access to the software and the project information

References

Acknowledgement

Jerome C. Glenn and John. J. Gottsman who, together with Theodore J. Gordon (chair) and Peter P. Yim, form the SOFI Advisory Committee that guides this work;

Douglas C. Engelbart, Adam Cheyer, David Leibs and Hal Hildebrand who contributed as members of the SOFI system architecture advisory panel;

SOFI system development team members: Adam Cheyer, Darren Haas, Eric Garlick, John Bear, Jonathan Cheyer (technical lead and architect), Kurt Conrad, Lisa Colvin, Mary Keitelman, Bo Newman and Peter Yim (project director).

I. HISTORY OF THE SOFI SYSTEM

The Millennium Project of the American Council of the United Nations University started publishing the State of the Future Index ("SOFI") time series, and the research and methodology behind it, in their 2001 State of the Future Report. Ted Gordon developed the methodology and the original algorithm used in the SOFI computation. (See chapter 22 in this series.) The World SOFI's that came with the 2001 through 2003 *State of the Future* Reports were developed and computed on electronic spreadsheets. This original implementation used Lotus 1-2-3 and its macros, and this approach was very cumbersome, time consuming and not at all conducive to wide spread adaptation and use of SOFI.

The Project

Because of these limitations intrinsic in the original implementation of SOFI, a much more flexible, version was imagined which could be accessed and used in other domains, such as countries, industries, or corporations. Furthermore if a more advanced system were to be available, it could serve as a common repository for the arrays of data and judgments that make up a SOFI. With proper safeguards, it could become a vehicle for combining proprietary with public data to draw insights not otherwise attainable. In July 2002, The Millennium Project Planning Committee decided to look into improving the software and developing a full-scale implementation of the SOFI methodology. A SOFI Advisory Committee established the MP-SOFI-SD Project (Millennium Project - State of the Future Index - Software Tool/System Development Project; hereafter, "MP-SOFI-SD" or, simply, the "Project") as a sub-project under the Millennium Project.

Phased Implementation

Three initial phases of the implementation were identified:

- Phase 0: Strategizing, planning, staging & requirements gathering.
- Phase 1: Initial software tool implementation and the launch of a community of practice around SOFI.
- Subsequent Phases: Continuous development and improvement of the SOFI system, content and knowledge through the collaboration of open communities that are augmented by the Internet and other technologies.

A Project kick-off meeting and the first architecture design workshop was held in the Silicon Valley on November 8, 2002 at SRI International (Menlo Park, CA, USA) and the Project was officially launched.

Core Concepts and Values

Of particular significance to this full-scale implementation of SOFI are the following concepts and processes, which, together, form the core values of the approach:

(1) The **SOFI Methodology** combines historical data and future forecasts, quantitative as well as qualitative techniques, mathematical derivation as well as human judgment, to systematically paint a picture of the state of the future. The SOFI time series data and their graphical representation provide highly condensed information and visual impact. But, also of great value to other users and analysts would be the details (historic data, assumptions, choices, judgments, debates, studies, inter-relationships, cross-impacts, etc.) that contributed to the resultant SOFI. For this reason it was important in this project that all of the information that contributes to a SOFI is properly captured, stored and made available through some knowledge access system. Current and future work in information, knowledge, artificial intelligence and collaborative systems do hold the promise of a continuously improving ability for this sort of application.

(2) A paradigm called "**Bootstrapping**," originated from visionary/inventor Doug Engelbart, which envisions the kind of new, co-evolutionary environment that can be created to help cope with simultaneous complex social, technical, and economic changes at the rate, scale and urgency that humanity is now faced with, is being adopted here.

Of particular relevance to the work here are the few (excerpt) ideas cited in Engelbart's Bootstrap mission statement [1]:

- Promote development of Collective IQ among, within and by networked improvement communities;
- Cultivate a knowledge environment which includes a shared dynamic knowledge repository;
- Foster development of an open platform information system infrastructure, based on an Open Hyperdocument Systems (OHS) framework;

(3) Serving a **wide** variety of **applications**. While the millennium project developed and used the SOFI concept to produce a global forecast of the state of the future, other possible applications illustrate the potential of the approach. Countries, regions, cities or communities could develop SOFI's based on factors that they themselves consider to be important in defining their state of being and expectations. If many countries were to develop their own measures, their improvement or degradation could be compared, one to another, on a rational basis. Corporations could produce, track and make decisions on the basis of a corporate state of the future; and competitive analysis could be enhanced in this way. Indexes of various sorts could be constructed and forecasted for various purposes such as an index of moral behavior in an organization or an index of technological prowess in a nation. The intriguing aspect is that this wide variety of applications can use a common computational framework, a common means of communication among all of the contributors of an index, and to some important degree, even common data pertaining to time series and the external events that can deflect the future course of those variables. This project was designed to provide such a facility.

(4) **Collaboration** among participants and stakeholders in the form of virtual communities. A diverse, distributed, international community with the infrastructure and track record of collaboration in future studies would provide an ideal startup environment to grow the system on. This turns out to be the type of things leaders and participants of the Millennium Project have been doing all along. The node-based structure of the Millennium Project organization, possibly with the addition of some early adopter organizations, could, together, be organically formed into communities of practice and provide that kind of ecology needed. Further augmentation by state-of-the-art technology, provided that they are non-intrusive, and that properly training is conducted, will promise to improve the result even more. The participatory involvement of stakeholders formed into communities, with the use or development of SOFI's as a focal point, and augmented by a shared electronic workspace that helps alleviate some of the limitations posed by geography and time, begins to give us a glimpse into the co-evolutionary bootstrapping environment that Engelbart alludes to;

and,

(5) The notion of "**Openness**" -- as in open systems, open standards, free and open source software, open content, open collaboration, open knowledge and open mind. The process solicits and builds upon open participation, open contribution and open sharing of resultant work. This mode of operation promises more robust software, better efficiency, greater participation, and a myriad of systematic advantages [2] over closed/proprietary systems and other modes of development/ production of the system. In this vein, it has been adopted that the software developed for this project shall be made available under the General Public License (GPL) software licensing arrangement [3].

The Project is attempting to adhere to the above concepts and processes and to gain the advantages they promise. With the belief that (i) futures research can lead to normative futures, and (ii) that augmented collaboration can help harness the collective intelligence needed to help cope with the complexities of building a generalizable SOFI computation and use scheme. This is where we begin – where, hopefully, we will develop an initial infrastructure and a process that results in a participatory SOFI, and through the participation, better infrastructure and processes, better research, better SOFI's and better futures.

II. WHAT IS THE SOFI SYSTEM APPLICATION

The full-scale implementation of SOFI involves taking the existing personal research and analysis SOFI tool and making it into a robust, internet-based, open, collaborative SOFI user and developer environment.

As of this writing, the first phase of the Project is underway. During this phase, the team (i) has established an initial architecture, (ii) is developing a functional code base, and (iii) will develop and deploy a community of practice with early adopters. This approach is intended to spur continued development and improvement/expansion of the market, the use cases, the model, the

methodology, as well as the software and the system, once the first phase is completed. The following is a description of this phase of development, where the system is developed as an application software (“SOFI application” or “the application”.) The application is hosted on a series of servers that run on the Linux operating system and provides for an open, standards compliant, scalable, fault-tolerant and secure system.

The Phase-1 SOFI Application

The SOFI application is a scalable, web-based application that allows users to view, create, and modify data from the State Of the Future Index. Users can also perform a “run”, or calculation of the index, based on a set of data. The calculations and data can be seen in multiple ways (as text, numbers, graphs, etc.). Users can choose to collaborate with others, or work independently. Or they may work in small groups for a while and then when ready, “publish” the results for the world to see.

The data of the application currently include historical data of variables, future values of the variables based on a particular type of curve fitting, weights of the variables, hypothetical events that impact the variables, and the amount of the impact on those variables. It is expected that additional types of data will be added in future versions of the software.

The calculations performed on the data are handled by the SOFI computation engine. Currently, two types of calculation algorithms are processed by the engine: (i) the original SOFI algorithm that aggregates the forecasts of the variables, and (ii) the Trend Impact Analysis (TIA) that accounts for the impacts of future events in the variable forecasts. It is expected that additional types of algorithms will be available in future versions of the software.

The system comprises "server" software components that run on an Internet Server, and a "client" software which runs on users' personal computers. The application has support for multiple types of clients. Currently, a client written in Visual Basic provides the dynamic, graphical view of the data and calculations, and a built-in web browser provides a dynamic, text-based view of data and calculations. The browser can also display any type of static information, such as documentation, pre-defined graphs, and other information. Future versions of the software are expected to include multiple types of clients, with individual clients focusing on providing more sophisticated views on specific areas or uses.

In the following sections, we will discuss the goals of the SOFI application, the architecture that implements those goals, and the components of the application. The components include the computation engine, web front-end, Visual Basic client, database, and the collaborative platform.

GOALS

The predecessor to the SOFI application, the SOFI spreadsheet, focused exclusively on the calculations. While well designed, it had some limitations inherent in the spreadsheet model. However, it also had some advantages that are well worth mentioning. The spreadsheet model excels in providing a clear, concise view of the data. The tabular format is easy for people to read, and is quite familiar to the average business computer user. Programs such as Lotus 1-2-3 and Microsoft Excel have done much to further the spreadsheet model and make it available to almost any MS Windows user. More recent programs such as OpenOffice provide a Free Software alternative to commercial products, and it works on multiple platforms such as Windows, Linux, Solaris, and Mac OS X.

Spreadsheet programs are simple to install and do not require complex administration skills. No programming skills are needed to use it, although some mathematical background and understanding of formulas is very helpful. However, spreadsheets are primarily written for a single user, working on a local computer (usually a PC). The “architecture” of the spreadsheet is usually described as a single tier, meaning that the presentation, the logic, and the persistence are all tied together in the spreadsheet. This provides advantages in simplicity, but it is not very flexible.

Spreadsheet formulas are handled on a per-cell basis, and the presentation of the cell is specified together with the formula. In order to apply a formula to multiple cells, the formula is usually copied to each cell, instead of referenced a single time. Both mistakes in copying formulas and mistakes in which cells the formulas are referring to, can be very difficult to locate (although easy to fix when found). Spreadsheets don't make clear distinctions between original inputs and temporary, intermediate results that are used as inputs later in an algorithm, since everything is represented by a cell. (Whereas programming languages can sometimes do a better job at providing a clearer abstraction with the introduction of various programming artifacts, such as local variables and data encapsulation.)

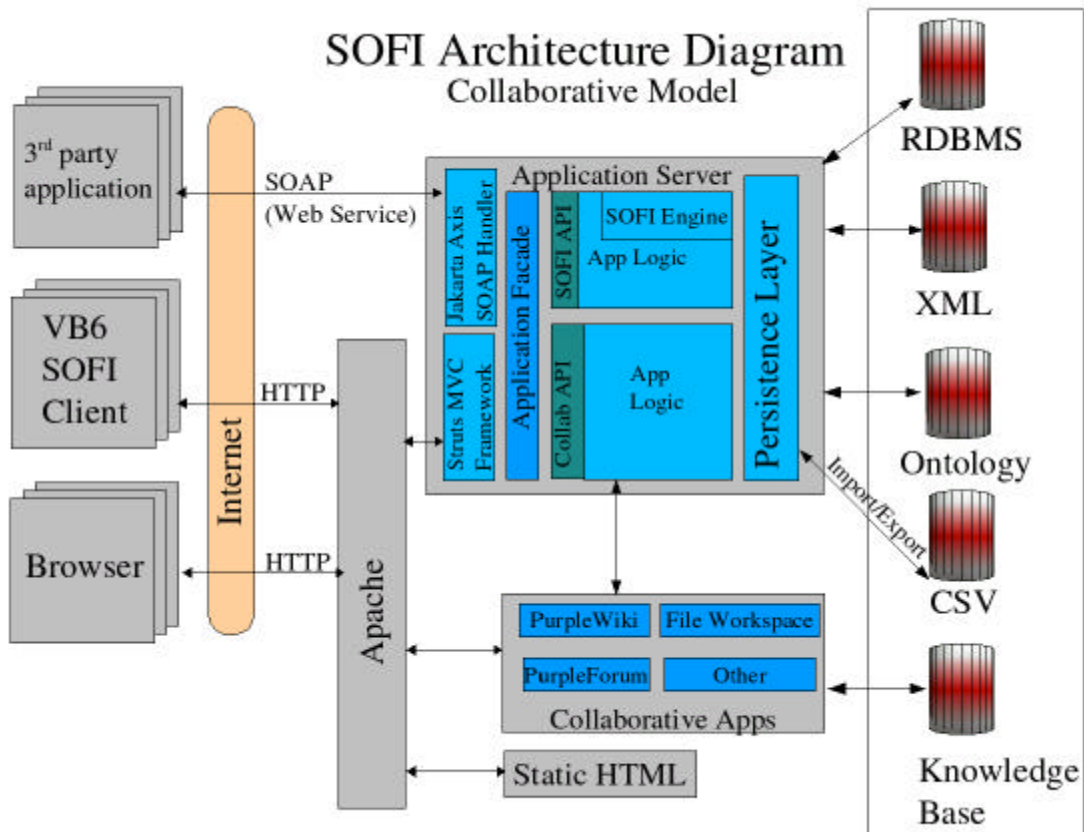
The SOFI application was written in response to some of these limitations. Creating a SOFI can be a complicated thing to do and take quite a bit of time. Data must be obtained, and sometimes massaged, to get it into the proper format. Weights must be determined. Events must be chosen, and their implications considered. All of this suggests that work could be done in a collaborative way, with peer review and discussion of the finer points an important aspect of the process.

The application should be able to be more than just a replacement application for the SOFI spreadsheet. It should support a SOFI community, one in which participants may be in more than one country, speaking more than one language. It should be able to provide support for the collaborative process itself, helping users work together. It should be able to support multiple ways of viewing data. It should be able to store its data separately from the application in a standardized way so that the data can be read even without using the application. The code itself should be open, available for users to inspect and improve.

ARCHITECTURE

The architecture of an application is what holds all the pieces together, and drives it towards the goals of the application. Figure 1 is the architecture of the system being implemented.

Figure 1: SOFI System Architecture



The language chosen for the current implementation of the SOFI application is Java. While no language is perfect, Java is a good fit for the implementation language for the SOFI application for a number of reasons. It is considered to be a modern language and has support for things such as object orientation, garbage collection, exception handling, and security. It has also been around long enough to be fairly mature at this point, with stable, well-published APIs and language constructs. There is widespread support both from industry and Free Software/Open Source organizations. It has strong type-checking and is a compiled language, which assists greatly in reducing bugs. Automatic variable boundary checking by the Java Virtual Machine and removal of direct address manipulation from the language render impossible the most common security problem found in C language programs: the buffer overflow error. Java runs on many platforms, including Windows, Linux, Solaris, and many others.

Java is especially good in web-based applications in which it is running on a server. For this reason, the Java 2 Enterprise Edition (J2EE) technology was chosen. There is a large number of

existing API's and frameworks written that eases the development effort for applications. J2EE allows for competing implementations of well-known interfaces, which gives developers the ability to choose the best implementation for their needs without losing interoperability and option of changing their choice later on. The Java Community Process allows companies, organizations, and individuals to participate in the process of deciding the future direction of Java and of the various standards and specifications.

A 3-tier architecture was chosen, following a well-known J2EE pattern. The tiers are broken up into presentation, logic, and persistence. This allows for the possibility of having different clients, each which can be plugged into the core infrastructure on its own. Because of the many ways in which data can be represented, a new client may be written that delivers the user interaction in a novel way. Likewise, there may be multiple schemes for the persistence, such as storing the data in a relational database or an XML file, for example.

The model and logic is exposed through a public, language-neutral, platform-neutral API known as a Web Service. The definition of the web service is described using the Web Service Description Language (WSDL). The protocol used to communicate with the web service is the Simple Object Access Protocol (SOAP), which is essentially XML over HTTP. Both WSDL and SOAP are open standards, defined by the World Wide Web Consortium (w3c). It has strong support from industry such as Microsoft and Sun, as well as open source organizations such as Apache. The current implementation used is Apache Axis, which is a SOAP processor that runs as a Servlet in the Apache Tomcat application server. By providing the core pieces of the SOFI application as a web service, other third-party applications could leverage the functionality of the SOFI software without it being seen by third-party application users as a separate application.

COMPUTATION ENGINE

The computation engine is at the heart of the application, as far as generation of the SOFI time-series is concerned. It is responsible for computing SOFI values from the given input data. The current engine can handle both the original SOFI calculations as well as TIA SOFI calculations. The engine defines and implements a public API. This allows for the possibility of replacing the existing implementation with a completely different engine implementation that performs the same tasks.

Although the engine normally runs in a J2EE application server within the SOFI application, it is intended to be able to run as a standalone Java process as well. This would allow other Java programmers to re-use the SOFI engine even if they didn't want to use any other portion of the SOFI application, such as the user interface or persistence.

WEB FRONT-END

Most of the current SOFI application is seen through the web front-end [B]. It interacts with the rest of the application, including the engine and persistence layer, to provide the user experience.

Since the user interface for the text is done through HTML, a simple web browser is all that is needed in order to participate in much of the collaborative work through the SOFI application. This allows the widest possible user base, since the requirements on the client software is just a web browser on any operating system, and some kind of TCP/IP network connection to the server. Normally, the server would be located somewhere on the Internet. However, it is also possible for a company or organization to run their own server, for example on an intranet, or even to run both the web front-end and the server on a single computer (although the installation and configuration is more complex than installing a spreadsheet program). The HTML is generated through a standard J2EE technology called Java Server Pages (JSP). The JSP pages are processed by a J2EE-compliant Servlet and JSP application server. By default, the SOFI application uses Apache Tomcat, an open source implementation by the Apache Software Foundation, as its servlet container/application server.

Standard JSPs provide a good separation of presentation detail (HTML) from Java code. However, an even cleaner approach to handling presentation is to separate the presentation processing from the presentation view. This involves a well-known pattern named Model-View-Controller (MVC).

The current SOFI application makes use of Apache Jakarta Struts, which is one of the open source frameworks that uses the MVC pattern, and is built on top of JSP. Beyond providing a cleaner abstraction, it also provides good support for internationalization (i18n standard compliant) and localization (l10n standard compliant). Furthermore, it offers some simple assistance to the JSP programmer in handling HTML form validation and error control.

VISUAL BASIC CLIENT

The user views the text and the graphical information seen in the SOFI application through its Visual Basic (“VB”) client. This VB client software contains a built-in web browser that is used to navigate through the various features of the application. It also allows the user to see graphical representations of the data along with their values, such as those for each of the variables, or for the computed SOFI.

DATABASE

All of the data in the application is stored in a relational database. By default, the SOFI application will use the open source database MySQL. However, since the application uses SQL calls (through the standard JDBC API) and does not use stored procedures or other database-specific logic, it is possible to use any JDBC-compliant database.

The data can also be exported in various formats, such as a comma- or pipe-delimited format that is common for spreadsheets.

An expected feature of a future version of the SOFI application will include the ability to export the data in XML format.

ONTOLOGY

Ontology is the standardization of meanings. An ontology defines the terms used to describe and represent an area of knowledge [4]. Ontologies are used by people, databases, and applications that need to share domain information. Ontologies include computer-usable definitions of basic concepts in the domain and the relationships among them. They encode knowledge in a domain and also knowledge that spans domains. In this way, they enable semantic interoperability and hence allow knowledge to become more reusable.

In developing the ontology for SOFI, descriptions for the following kinds of concepts will need to be captured:

- Classes (general things) in the many domains of interest
- The relationships that can exist among things
- The properties (or attributes) those things may have

The reason why ontology is designed in as part of the SOFI system architecture is that ontologies figure prominently in the emerging Semantic Web as a way of representing the semantics of documents and enabling the semantics to be used by web applications and intelligent agents. Ontologies can prove very useful for the SOFI community(ies) as a way of structuring and defining the meaning of the data and metadata terms that will be collected and standardized. The ontology, and the ontological engineering approach adopted, and to be implemented in future phases, provides the basis for the system's knowledge architecture. This will enable the SOFI system to support tomorrow's "intelligent" application, giving its users better access to the information and knowledge that is continuously being captured into the system as it is being used.

COLLABORATION PLATFORM

As the SOFI application aims to support collaborative work by the SOFI community, it is important to provide tools to enable or facilitate collaboration. The phase-1 application will leverage the features found in some of the best web-based, open source tools as integrated in the CIM3.NET Collaborative Work Environment suite. The platform will include both tools and processes for asynchronous and synchronous collaboration support. It features:

- an archived e-mail/discussion forum,
- a community wiki (which is a website that is both writable as well as browsable),
- a document repository / file-sharing workspace, and
- a web portal. This portal is accessible from the SOFI application web front-end, as well as from any web-browser,
- support for full-text search and fine-grain (paragraph-level) access to shared content.

Synchronous collaboration support include:

- voice conferencing,
- screen/application sharing,
- instant messaging, and
- real-time chat session

The primary mechanism in which the collaborative applications communicate with one another is (through TCP/IP) over the Internet. The most popular user applications are retained as the interface to the user, thus lowering the learning curve for new users. As an example, user discussions are made through electronic mail with whatever e-mail system a user may already be using. The e-mail messages exchanged are then processed, transparently, by the system -- archived, parsed, turned into web pages, indexed (to facilitate listing and search), ... -- so that any user in the community may retrieve or cite an earlier message (or parts of it) with his/her web browser.

Some of the primary software used currently includes forum list management (Mailman) and archiving (mharc), search (namazu), and community-writable web site with paragraph-level accessibility (UseModWiki / PurpleWiki). Fine-grain accessibility (Purple / PurpleSlurple) is achieved through the use of various tools that implement the Engelbart/Bootstrap/OHS notion of "purple numbers"[5].

It is worth mentioning that that the emerging WebDAV standard is adopted here. Briefly, WebDAV stands for "Web-based Distributed Authoring and Versioning". It is a set of extensions to the HTTP protocol which allows users to collaboratively edit and manage files on remote web servers. With that, users can be sharing electronic files with drag-and-drop ease from almost any personal computer (whether it runs on Windows, Linux, Unix, or Macintosh system.) Optional mechanisms are in place to prevent users from overwriting each other's work. The promise of true versioning in the future development of WebDAV will enable even greater system flexibility and capabilities.

With this collaborative platform, distributed SOFI developer and user communities will be able to carry on discussions, interact, learn, share, provide mutual support, explore, innovate and thrive as communities.

III. HOW TO USE THE SOFI SYSTEM?

The Millennium Project has intended for this to be a simple-to-run, integrated, program with a well-designed graphical user interface for calculating SOFI. The value proposition for such a tool is best expressed in Chapter 2 of the 2001 State of the Future:

“The SOFI is a statistical combination of key global indicators and forecasts that depict whether the future promises to be better or worse. If the promise of the future seems to be changing, then the SOFI is intended to show the directions and intensity of change and to identify the factors responsible. If confidence were developed in such an index, it could be used in policy analysis at the global, national and local levels. Furthermore, nations could be ranked by their SOFI to determine if the future outlook in their regions was better or worse than the global outlook and the reasons for any differences.”

Who are the users?

The target audience for this tool are those people interested in the SOFI, and, in particular, those who want to develop and compute their own SOFIs. The latter could include:

- Global, regional and corporate policymakers
- Professionals, scholars, academics
- Economists, political scientists, social scientists, etc. , and
- The Millennium Project and its Nodes

Users will need to work with computers (or have staff support that does) and will need to have access to the Internet, to take full advantage of the system functionalities.

We can also identify different classes of users. Each class of user will look at and work with the system differently. These classes of users may include:

- General public
- Researchers, journalists, trend watchers
- Policy makers
- SOFI developers, modelers, futures researchers
- Data owners (those who own and maintain databases whose data is used to construct the SOFIs)
- System developers (those who will contribute to the software)
- System Administrators (those who administer the SOFI system)

What can we do on the system?

Users will be able to do some (as this is only a partial list) of the following: (Note that some functions will require the user to be set up with proper access rights and authentication.)

Viewing/reading up

Any user accessing the system will be able to view all SOFI's (current as well as archived historic versions) that have been published for public viewing. The Millennium Project's published Global SOFI is in this category. There are also SOFI's (particularly works-in-progress) that may be restricted (through password authentication requirements) to be viewable only by certain communities, or even by the authors alone.

In addition, the document repository will host, among other contributions of the SOFI communities, all documentation relating to the creation and the analysis of the public SOFI's (for example, the SOFI chapters in the MP State of the Future Reports or this Futures Research Methodology v2.0 will be accessible there.) Therefore, users will be able to read up about the SOFI's and how they are constructed as well.

Comparisons

When multiple versions of a SOFI (e.g. MP has published the Global SOFI for 2001, 2002 & 2003) are available, a user will be able to call up more than one SOFI time series, and compare them (charted with each represented as a different color plot). Similarly, if someone had already done and published the country SOFI for, say, the US, the EU and China, one will be able to pull them up, and overlay them on top of one another, and compare them with each other and with the Global SOFI of that particular year.

Drill-downs and searches on related information

This system is implemented along the lines of Engelbart's Open Hyperdocument System ("OHS"). One can think of all the data, information and knowledge as being connected and accessible through a web of hyperlinks. Unlike the World Wide Web as we know it now, where access is essentially limited to the page level, hyperlinking within the SOFI system is done at a finer-grain level (at the paragraph level for text documents and archived discussions, or at the data-series level, for example). Therefore, not only will users be able to read up about the SOFI and related material, they will also be able to drill-down from one level of information to the next lower one (in most cases).

It would be dull or exciting- depending on one's point of view- to learn that from year 2007 to 2008, the SOFI improved from 1.0634 to 1.0724, or that it drops back to 1.0691 in year 2010,

but more importantly would be for one to find out what variables, which datasets, what assumptions were responsible in making the index change. If one is interested in a particular variable, one could go on and discover the associated Delphi study and the opinion of the panel of experts in events or developments that affect this variable. The non-linear construct of the hyperdocument will allow the users, simply through mouse-clicks, to drill-down deeper and deeper to find out why.

While more sophisticated knowledge access structures will be implemented in future phases, the phase-1 implementation supports full-text search, and provides us, yet another means of information and knowledge retrieval and discovery.

Asking what-if questions by “tweaking” the SOFI parameters, or exercising alternate Scenarios

Policy makers will have the option to “tweak” some of the SOFI parameters – like, forcing probabilities of a certain event happening in certain years to 1 (or 0) -- and seeing how the SOFI changes. That in effect, allows the policy maker to ask a question like: “what-if ‘Mad Cow disease found in many countries’ happens (for sure) in year 2003 and 2004, how would the future outlook for the next 10 years be affected.”

Another option is to adjust the weights applied to a (or a few) variable from the value assigned by the publisher of the SOFI.

Analyst, researchers and policy makers may even construct alternate SOFI’s based on different scenarios, and see how those SOFI’s will be alike or differ, and how they compare to a baseline SOFI.

Developing new SOFIs

A major purpose of the system, of course, is to facilitate the development of new SOFI’s. Examples would include:

- SOFI’s for different Countries, Regions, Municipalities, ... etc. (e.g. a USA-SOFI, France-SOFI, Russia-SOFI, China-SOFI; or a EU-SOFI, Middle-East SOFI, Greater-China SOFI, Latin-America SOFI; ... etc.)
- SOFIs for different government or industrial sectors (e.g. E-Government SOFI; Petroleum and Oil industry SOFI; Automotive Industry SOFI; Aerospace industry SOFI; Science & Technology SOFI; Nanotechnology SOFI; Tertiary Education SOFI; ... etc.)
- SOFI for Corporations; individual Organizations; Programs, Initiatives or Product Lines; ...etc.

One should look at SOFI as a methodology that allows one to aggregate relevant historical and forecast data, and summarily quantify it in a form that depicts whether the future outlook is

improving or regressing. Through the processing of both statistical as well as judgmental data, one can see that it provides a means and a generic tool and process for people to get a grasp on highly complex and fuzzy situations.

While no one can predict the future, the construct of this system, which facilitates the evolution of the process and knowledge captured in the system, and with its attempt to leverage the collective intelligence of communities of practice, promises a continuous improvement on its effectiveness as a policy and decision support tool.

Collaboration and Collaborative Development

The collaborative platform provides the users with the means to (a) use/re-use data, information and knowledge that have been captured into the system, (b) to communicate, seek help and support from fellow practitioners, and (c) to engage in community-wide discourse, plan initiatives, and coordinate projects and team work. It provides the workspace for community members to share their work, their knowledge and wisdom with one another.

The primary user interface is through the SOFI client software, the web browser, a user's e-mail, and/or through drag-and-drop operations into community folders (which, for today, should not be presenting much of a challenge to most users now.) Archival of the discussion and/or contributions into the system repository is made almost transparently. All contributions become sharable (if so intended) by the community. They will also be retrievable through the system search engine (which, as mentioned before, supports full text search) and be made accessible at a paragraph level granularity.

As described above, real-time online meeting (like phone conference with shared-screen) support are also available. The system and process design that makes up this full-scale implementation, is optimized for the collaborative development and application of SOFI, by distributed communities of practice and project teams.

While ease-of-use and minimal intrusion is built into the design of the system, it is recognized that proper training for users of the system will greatly enhance its effective use. This will be especially useful for users who would want to make the best use of virtual collaboration tools to perform distributed team work. The entry barrier is intentionally kept low with a mild learning curve for the beginning users. Adept users, however, will find it equally rewarding once they start to appreciate the power and performance offered by the more sophisticated features of the system.

Interaction and accessing data, information and knowledge

The system is design to support three kinds of interaction:

- Human-to-machine
- Human-to-human
- Machine-to-machine

We have described much of the first two types of interaction in earlier paragraphs within this section. It is useful to mention that the system is also designed with machine-to-machine interoperation in mind. Some features (like the employment of web services as the means to handle the interaction between the user's machine and the application server; or the systems ability to import and export data that are compliant with certain standards, like CSV formatted data) are implemented in the phase-1 implementation, while others are planned and designed into the system architecture (like the ontology and associated knowledge access structures), to cater for the future, as the Internet and other system and computer technologies continues to evolve.

Building up the system's database and knowledgebase

For a SOFI developer, every time a set of data is required (e.g. the historic figures for GDP per capita between 1980 to 2002, or the judgmental input of a panel providing normative and dystopic values and weights to the development of "Biotech in agriculture"), he or she will either have to (i) retrieve it from the system database (if it is already in there, exactly the way it is needed), or (ii) take what is in the database, and update it (if it is already there, but not updated), or (iii) source a set of new data and enter it (manually, semi-automatically or automatically) into the SOFI system. If this developer is sharing his/her data with the rest of the community, the sharable database grows by that much, whenever updates are made or when new data are added. The same thing goes with discussions and work contributed to the wiki or the file workspace/document repository. As such, the database and captured information/knowledge grows as the system gets used.

The chances of the reuse of previously captured data depends on whether another user has had a use for it, and if so, has made an attempt and successfully retrieved it. The system is designed with ease of data and knowledge capture and retrieval in mind. That will be one area that continuous improvement is sought.

That said, the content of the database (of numeric data, time series, and SOFI datasets) and the captured user collaboration (from their discussion forum, wiki and contributions to the document repository) will, collectively, form the system's knowledgebase, or, in Engelbart's terms, our Dynamic Knowledge Repository ("DKR"). This knowledgebase is expected to grow as the system gets used. For example, different SOFI's employ different sets (even if some are common) of variable data for their construction, and that contributes to the growth when more people use the system to construct different SOFI's. Likewise, when more people participate in the

communities, there will invariably be more expertise within its membership. One could, then, expect a better likelihood to get a good answer, say, when one has a question to ask of the community. Since shared knowledge are captured, archived and retrievable, that builds up too with use and participation. With a more substantial database and knowledgebase, users will be able to make better analyses or syntheses, and different SOFI's can be constructed more easily. With that, more people will be attracted to participate in the community(ies) and use the system. This is the “bootstrapping”, and the kind of virtuous circle that the authors wish to see happen.

IV. STRENGTHS AND WEAKNESSES

Most of the details are already covered above. In summary, the salient strengths and weaknesses and challenges of this full-scale implementation of the SOFI include:

Strengths

- Ability to drill-down
- Robust – built to enterprise-class quality standards (not as an academic experiment)
- Accessible & Available – by way of the “openness” core value adopted, and because it is hosted on the Internet
- Distributed – designed for people to work from different locations
- Collaborative
- Organic: supports “bootstrapping” and co-evolution

Weaknesses

- Needs some level of sophistication in the part of the users
- Complexity (despite all efforts to lower the entry barrier to users)
- Learn probability – while the system is already hiding most of the mathematics, an understanding of how the system and its computation works is essential for proper application
- Needs training
- Needs computer and Internet access

Challenges

- Maintenance of the data (even historical data change)
- Building and nurturing early adopter communities to the point when critical mass is attained

V. USE WITH OTHER FUTURES RESEARCH METHODS

The system described here can both provide a service to other applications and draw from other methods to facilitate its operation. On the service side, this system could provide a source of information for applications that need quick access to time series data on variables that relate to the state of the future. Such applications include, potentially, simulation models, gaming, and econometric applications. These ties would be particularly strong and important if the database in SOFI were updated periodically and were known as the source of high quality, accurate global information. Today, some commonly available and respected sources provide data only for selected countries; the SOFI database could extend this piecemeal information to an authoritative global estimate and in that sense become unique. The database associated with the TIA's could also be important to other applications. For example, one could imagine a simulation game in which the TIA events provide a set of "discontinuities" that are superimposed on an otherwise extrapolative series of moves. Policies could be tested for robustness against the surprises that the TIA events represent. The TIA database could be the source of exogenous events for econometric models.

The system can make use of several futures research techniques in its own construction and operation. For those elements that require expert opinion and judgments, Delphi in one form or another could be employed. Public opinion polls could also be used to identify the variables that are important in the "public mind" and these could be compared to expert Delphi assessments. Advanced scanning techniques could be used to monitor the appropriate source to determine if new events should be added to the database or removed as a result of their having occurred. Finally, there are TIA event impacts that might be computed, rather than judged, using various modeling techniques such as econometric modeling.

It would be interesting to compare the SOFI forecasts with opinion polls about the future. Both, after all, deal with perceptions about whether things appear to be improving or deteriorating. We suspect that opinion polls will be much more volatile and responsive to the problems of the moment, but a comparison of this sort might provide a means gaining an understanding of public opinion about the future and the behavior it engenders.

VI. SPECULATIONS ABOUT FUTURE DEVELOPMENT OF THE SYSTEM

From the software engineering perspective, the continued development and adoption of some of the Internet trends and open standards embraced by the SOFI system architecture (like XML, WebDAV, SOAP, Web Services, DKR, OHS, the Semantic Web, ...etc., or some future incarnations of these technologies) will allow it to better interoperate with other systems, both as a provider and a recipient of services. For example, the SOFI application already exposes itself as a web service using SOAP. In the future, there may be other services, like those mentioned in section 5 above, that the SOFI application itself could make use of, in which case it would be a client of those services, while at the same time providing a service of its own.

On its application, if SOFI becomes more broadly used, countries and regions could track their own SOFI. Newspapers and other publications could cite, or even syndicate, SOFI as they would today with some of the economic indices (such as GDP, cost of living, DOW, and S&P500). Future phases of the software could associate SOFI with scenario building and validation, as well as with front-ends that facilitate the capture of judgmental or other research inputs. By architecting it as an open knowledge system, and developing this software in an "open source" mode, it is envisioned that it will become a collaboratively developed and continuously improving system. Among others, is the normative desire to maintain an authoritative, accurate and up-to-date database of data relevant for computing SOFI's, and to make such database available in a consistent, standard format. Coupling that with the research content and scenarios, the SOFI software tools and system shall, hopefully, become a useful and compelling tool/platform/knowledgebase for the futures research community, policy makers, strategists, scholars and other SOFI users.

The Millennium Project, along with SOFI developer and user communities, could develop and establish quality standards, and even the trust system that would identify published SOFI's that are developed with stringent process control and quality assurance, and to properly recognized them to be deemed trustworthy.

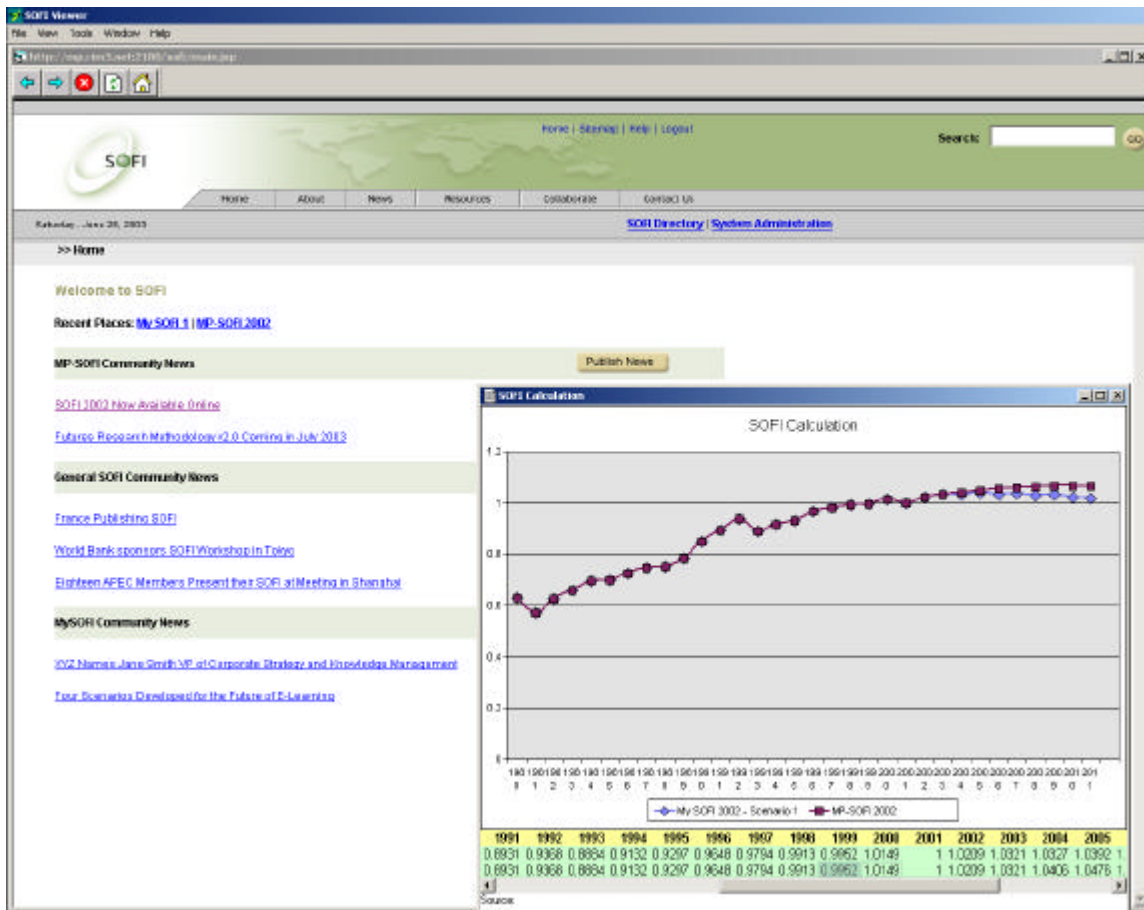
The full-scale implementation of SOFI is envisioned as a multi-phased endeavor of which only the beginning phases are described here. What is being worked on is a system that is designed to provide a work environment conducive to organic growth of knowledge and human interaction, so that people can collectively work out how the future may look like. Let us hope we will see some emergent pattern and behavior the next time we look again.

APPENDIX

Access to the software and the project information

[A] Portal to the MP-SOFI-SD Project information is at:
<http://mp.cim3.net/file/project/mp-sofi-sd/home.html>

[B] Figure 2: an incipient phase of the SOFI client software interface and the computed SOFI (mock-up)



References

- [1] Engelbart, Douglas C., "Bootstrap" Vision and Mission (Dec. 1999)
<http://www.bootstrap.org/ba/index.jsp#nid01>

- [2] Coase's Penguin, or Linux and the Nature of the Firm (Jan. 2003)
<http://www.benkler.org/CoasesPenguin.html>

- [3] The GNU General Public License (Version 2, June 1991)
<http://www.opensource.org/licenses/gpl-license.php>

- [4] Web Ontology Language (OWL) Use Cases and Requirements (Draft, March 31, 2003). See:
<http://www.w3.org/TR/2003/WD-webont-req-20030331/#onto-def>

- [5] See <http://purpleslurple.cim3.org>