Quality of IT Service Delivery - Analysis and Framework for Human Error Prevention

L. Shwartz, D. Rosu, D. Loewenstern, M.J.Buco,
S. Guo
T.J. Watson Research Center, IBM,
Hawthorne, NY, USA
{lshwart, drosu, davidloe, mjbuco, sguo}
@us.ibm.com

R. Lavrado
Math, Computer Sciences and Engineering Division
KAUST, Thuwal, Saudi Arabia
rafael.lavrado@kaust.edu.sa

M. Gupta, P. De, V. Madduri, J. K Singh
IBM Research India, Delhi
{gmanish, pradipta.de, madduri, jaksingh}
@in.ibm.com

***Abstract*—In this paper, we address the problem of reducing the occurrence of Human Errors that cause service interruptions in IT Service Support and Delivery operations. Analysis of a large volume of service interruption records revealed that more than 21% of interruptions were caused by human error. We focus on Change Management, the process with the largest risk of human error, and identify the main instances of human errors as the 4 Wrongs: request, time, configuration item, and command. Analysis of change records revealed that the human-error prevention by partial automation is highly relevant. We propose the HEP Framework, a framework for execution of IT Service Delivery operations that reduces human error by addressing the 4 Wrongs using content integration, contextualization of operation patterns, partial automation of command execution, and controlled access to resources.**

Index Terms— Change, Human Error, IT Service Support and Delivery, Partial Automation

I. Introduction

Human error is one of the major causes of failures in today's IT environment, impacting systems from personal computing to global IT Service Support and Delivery operations. Different from technical failure, human error is more difficult to model and prevent. Due to the high potential of business impact, organizations and research are giving increased attention to understanding the causes of human error and finding effective methods for preventing it.

Root cause analysis and human error are areas of continuous focus for high risk industries such as nuclear power and aviation. Multiple studies have been conducted with the goal of understanding the types of mistakes people make in order to establish barriers to prevent people from making these mistakes. These industries are among the safest. Manufacturing and, to some extent, services have been focusing on Six Sigma and Lean Six Sigma as a structured approach to root cause analysis. However, Six Sigma and Lean Six Sigma rely heavily on the use of statistics and logical planning and decision making, and address only a small subset of human errors through use of Poka-Yoke techniques.

The focus of our research is on limiting the occurrence of Service Interruptions (SI) caused by human error in the context of IT Service Support and Delivery operations. The difficulty of the problem stems from the characteristics of work specific to IT Service Support and Delivery. Our own observations and previous research [6,8,10-13.20] highlight the high complexity, risk and stress of System Administrator (SA) work in IT Service Delivery operations. Complexity derives from task complexity [8,12,20], concurrent task execution [8,11], interaction with a multitude of tools and systems [8,11], and distributed knowledge across peers and teams [6,12,14]. Risk and stress derive from the mission-criticality of the supported systems and applications and contractual agreements and their penalties for Service Level Objective (SLO) failures. Such complex work conditions lead to numerous instances of error, spanning from simple command typos that cause the shutdown of the wrong server to incorrect specification of a change request that leads to failures of change deployment and SLA penalties.

Our study is based on observation and analysis of operations in a very large IT Service Support and Delivery organization, through shadowing SAs in their daily activities, conducting interviews, and analyzing documents related to service interruptions related to support and delivery of IT services. We shadowed SAs over a period of six months and conducted interviews with multiple Subject Matter Experts (SMEs) in various locations. Our data analysis, based on documents related to almost a thousand service interruptions, revealed that human error can occur in any process and any component of the IT Service Support and Delivery operations, including help desk, incident resolution, release, and change implementation. A significant percentage of service interruptions are attributed to human error based on results of root cause analysis, Further, we hypothesize that the percentage is even larger as almost half of SIs have incomplete root cause analysis.

In this paper, we address the problem of reducing the occurrence of human error in IT Service Support and Delivery operations. The scope of our work includes any type of activity performed on the targeted IT systems as a part of ITIL: compliant processes, such as change implementation, backup triggering, health checks, etc. Our analysis of service interruptions showed that even a simple health check could lead to service interruption if executed incorrectly. The goal of our work is to design and implement a framework for

execution of server/configuration item operations that can reduce significantly the likelihood of human error. We work to determine the framework elements that are necessary to achieve this goal and specifically focus on change management, the process with the largest risk of human error based on our study and related work [8, 10-13].

Change operations affect all the components in an IT environment, such as servers, networks, databases, and applications, with granularities that range from modification of a single OS configuration parameter to delivery center relocation. In many instances, changes have a high risk for the enterprise, where the smallest human error can cause widespread service failures. From the study of numerous instances of human error and discussions with SMEs, we identified four types of functional manifestation of human error that lead to erroneous execution of change and release operations: wrong request, wrong target, wrong time, and wrong command. These four 'wrongs', henceforth named the 4 Ws, guide our search for novel techniques for reduction of human error occurrences.

Towards this end, we analyzed over 200,000 change requests performed over 2 years in our target IT Service Delivery organization in order to understand the type of change actions that are performed most frequently. For the identified categories, we considered the applicability of automation, a method proposed by previous research for reduction of human error for anticipated tasks [1]. Overall, the analysis revealed that full automation can be used only for a relative small share of changes. However, partial automation, with an intertwined operation of the SA and an automation script, can be used for a reasonably large share of changes. Only very few of the changes could not be automated to a relevant extent, such are the hardware changes.

Based on these insights, we propose the HEP Framework, which comprises the elements to help reduce the occurrence of each of the 4 Ws. First, to address 'wrong request', the framework mandates tools for catalog-based specification of change requests and customization based on organization or workgroup specific parameters. Second, to address 'wrong time' and 'wrong CI', the framework includes tools that moderate the access of the SA to the target CIs at approved times. Third, in order to address 'wrong command', the framework includes tools for Human-Supervised Change Implementation, namely the execution of change implementation scripts in a partially automated approach. The role of the SA is to assess the correctness of automated execution of change actions based on automatically customized scripts. As a result, we submit that the likelihood of human error diminishes significantly.

The work presented in this paper differs in many respects from previous research related to human error in general and to IT Service Delivery in particular. Human Factors research developed a multitude of models for characterization of human error, taking into account a large variety of factors such as cognitive, procedural, and organizational. Our work characterizes human error with respect to the functional impact as an enabling step for identification of best strategies and methods for error prevention. Related to IT Service Delivery, previous work has characterized the complexity of SA activity and proposed tools to support the SA activity in general, aiming primarily to improve productivity. The framework proposed in this paper is specifically focused on change and release management and integrates a multitude of 'Wrong'-driven methods in order to provide good coverage for reduction of human error,

Another contribution of the paper is the novel categorization of change operations with respect to the action type, based on a very large volume of tickets. This work brings valuable insights to change management research community.

The paper is organized as follows. Section II discusses our analysis on the occurrence and factors for human error and service interruption. Section III presents the analysis for categorization of change activities and assessment of automation potential. Section IV presents the architecture of the HEP framework for reduction of human error in change management. Section V presents related work and Section VI presents our conclusions and future work.

II. Human Error Occurrences and Prevention of Service Interruptions

In this section we describe an analysis of service interruption (SI) incidents and the hypothesis that we derived from its results. Overall, service interruptions do not occur very often in the large IT Service Delivery organization targeted by this study. The study is based on a thousand SI records gathered over a period of twelve months. Each record contains details about the root cause analysis (RCA) of the SI and a SA marking of the cause, which may reference human error. Due to privacy concerns, a detailed description of the data is not provided.

Our analysis of the SI RCAs revealed that 53% of SIs did not have sufficiently detailed or complete RCAs to enable us to conclude with confidence the relationship to human error. In the SA marking of the cause, only 3% of SIs were marked as related to human error. Conducted surveys revealed that there is a significant difference between number of errors that caused by human and SA markings. It suggests that SAs have the tendency to classify human-related errors under other categories, mainly because of the ambiguous taxonomy used for classification in their organization. This finding led us to hypothesize that some of the 53% SIs with incomplete RCA were also caused by human error.

We found existing quantitative measures, such as error rates, to be inadequate and even misleading without additional information. Detailed and consistent descriptions of the conditions that led to human errors are critical for understanding the incidents and for driving the proposal for how to avoid future instances. In our case, we found no pattern to the conditions leading to human errors. Neither the user nor the type of change leading to the SI correlated with error rate. Therefore, we established a working hypothesis that the occurrence and frequency of human errors depends more on the interaction with the environment than on any stable, inherent characteristic of the SA or of the task. A methodology for Human Error Prevention has to be based on a theory of the interaction between human performance variability and the situational constraints.

We hypothesized that the rate of critical human errors in IT Service Delivery could be substantially reduced through changes in environmental factors. This hypothesis drew from a retrospective analysis of incidents. As part of this analysis, we analyzed RCAs and SI investigations and identifed causes in order to extract actionable proposals for a Human Error Prevention framework. For instance, we

considered the current training procedures and derived augmentations necessary to avoid human errors caused by inadequate training in the future.

Further, we performed a partial prospective analysis on the evaluated data and errors and analyzed the exposure to risk as part of a feed-forward loop. We propose that future work concentrate on prevention of SIs caused by four types of human errors – 4 W's: wrong request, wrong target, wrong time and wrong command.

III. Analysis of Change Activities

This section presents an analysis of over 200,000 change records collected over two years at a very large IT Service Support and Delivery organization. The goals of our analysis were the categorization of change activities and assessment of their potential for automation. We considered two types of automation: full and partial automation. We marked an activity for full automation if it could be performed completely without human intervention after being triggered by SA (autonomic execution). An activity was considered to be partially automated if its execution required an SA's intervention in one or more of the execution steps. For example, a patch installation that was triggered by an SA and completes without any additional SA actions was a fully automated activity. A database update that involved the SA running diagnostics and providing some of the data collected to an update scripts was a partially automated activity. A change activity was 'manual' if one could not automate it in given execution environment. For example, adding a hard disk to a system would be manual task for a physical system. Although the same task could be automated in a virtual environment, we marked this activity as manual (not automatable) because it required manual labor in the environment for which this request was received.

This section starts with a description of the 'change ticket', which is the data descriptor for a change operation used in the change management process of the target organization. Next, this section presents the method and the results of our classification of change operations. Finally, this section presents an analysis of the applicability of partial automation.

A. Data Model

In the change management process, details about the content and execution of every change request are represented as a 'change ticket' and stored across multiple database tables. Table I illustrates a small subset of the attributes of a change ticket. For instance, 'description' captures the free text description of the request with details about the high-level operations to be performed and references to the configuration items (CIs) involved in these operations. Similarly, 'type' captures a business-specific categorization with respect to the type of components involved in the operation, such as Application, Software, Network, Hardware, Environment and Operation. The 'risk' attribute captures a business-specific categorization with respect to disruption that the execution of change implementation can have on the overall activity. The change ticket also captures a characterization of the request completion, such as 'installed' (*i.e.*, action performed), 'partially installed', 'canceled', or 'back-out'.

B. Model of Change-Related Activities

The goal of our analysis of change tickets was to create a model of the activities performed by SAs related to change implementation. We modeled an activity as a pair of (1) type of action, such as 'configuration change', 'restart', 'reboot', 'update', 'relocate', and 'maintenance', and (2) type of CI, such as hardware, software, OS, database, application.

TABLE I
Sample Attributes of a Change Ticket

Attribute	Value
CHANGE_ID	127550
DESCRIPTION	Change the password on the probe
COMPLETION_CODE	INSTALL
TYPE	SOFTWARE
RISK	MINOR
START	03/01/06 11:04 AM
CLOSE	03/01/06 01:04 PM

Note that typically each activity type is sub-divided into lower-level activities. For instance, a 'configuration change' action for a 'database' CI comprises the following steps: (1) determine access details for the database, (2) connect to the database, (3) verify the current status of the target configuration parameters, (4) change configuration parameters to the requested values, (5) verify that the changes are applied, and (6) verify that the system is performing properly. For the purpose of this study, we did not target the classification of the lower-level activities, yet we used related knowledge to assess what types of human error could occur in change implementation.

TABLE II
Change action Categories

Action Category	% of Tickets
DB OPERATIONS	11.09%
FIX - CHANGE	9.87%
INSTALL - SOFTWARE	8.00%
UNKNOWN	7.67%
UPGRADE	7.57%
SECURITY – NON PATCH	6.29%
UPDATE - ADD	5.91%
UPDATE - REMOVE	3.91%
REBOOT	3.28%
UPDATE - FIX	3.28%
MAINTENANCE - SOFTWARE	3.22%
DB – UPDATE	2.94%
UPDATE - RUN	2.54%
RELOCATE	2.37%
INSTALL - HARDWARE	2.27%
UPGRADE - PATCH	1.98%
APPLY	1.69%
UPDATE - REPLACE	1.51%
UPDATE - CREATE	1.48%
MAINTENANCE - HARDWARE	0.79%
CLOSURE	0.56%
BACKUP	0.43%
UPDATE – CONFIG	0.31%
SECURITY - PATCH	0.28%

Extensive analysis of the set of change ticket revealed that structured attribute values do not provide enough information to model the change activity to a sufficient depth to be useful in predicting likely human error types. As a result, the basis for our classification was the free-text value of the

'description' attribute. Initial trials of using unsupervised methods and supervised methods with small number of labeled records resulted in poor accuracy because the ticket language features a high volume of non-significant words, (*e.g.*, product and CI identifiers, technical terms), abbreviations, spelling mistakes, and multiple languages.

TABLE III
CI TYPE CATEGORIES

Change Type Attribute	CI Type Category	% Of Tickets
APPLICATION	SOFTWARE	42.25%
SOFTWARE	SOFTWARE	31.70%
NETWORK	SOFTWARE	13.12%
HARDWARE	HARDWARE	6.95%
ENVIRONMENT	HARDWARE	3.76%
OPERATION	HARDWARE	2.19%

Eventually, we adopted an incremental computer-assisted manual classification based on keyword matching that created a change action categorization of 44 action categories. At the end of the classification work, 7.6% of the data remained unclassified ('UNKNOWN'). Table II presents the change action categories in order of ticket percentage, limited to 23 categories with a percentage of over 0.25% due to space limitations. The remainder of our change analysis was based on the 84% of the records in these 23 categories.

The change tickets do not have an attribute that explicitly describes whether the change is a hardware or software operation. We derived this operation category from the change type attribute as shown in Table III.

C. *Correlation of Change Category and Automation*

TABLE IV
SAMPLE OF HARDWARE-RELATED CHANGES

Description	Action Category
Installation of new IPM printers	INSTALL - HARDWARE
Replace broken disk	MAINTENANCE - HARDWARE
Move out two servers from A	UPDATE - REMOVE
Memory Install	INSTALL - HARDWARE
Move Fiber Cabling from A to B	RELOCATE
Replace hardware	MAINTENANCE - HARDWARE
Power cable cleanup	MAINTENANCE - HARDWARE
Power install on 1st floor	INSTALL - HARDWARE

The goal of this analysis step was to estimate the fraction of the change requests that we predict can benefit from partial automation. Our approach was to assess which changes cannot be (partially) automated to a reasonable extent, and which changes can be fully automated. The estimation was based on few observations. Regarding the changes that cannot be automated, we observed that hardware-related changes, such as moving cables or replacing broken hardware, require human manual operations, which make it hard to achieve a sufficient degree of partial automation of the change implementation to reduce the likelihood of human error. Table IV presents a few examples of hardware-related changes

Based on the operation categorization (Table III), we concluded that the hardware-related changes are about 13% of the change volume.

TABLE V
CATEGORIES WITH HIGH POTENTIAL FOR FULL AUTOMATION

Category	% Automation
INSTALL - SOFTWARE	75.00%
REBOOT	69.30%
UPDATE - FIX	96.00%
MAINTENANCE - SOFTWARE	95.00%
UPGRADE – PATCH	82.17%
BACKUP	86.32%
UPDATE – CONFIG	86.82%
SECURITY - PATCH	85.82%

Regarding the changes that can be fully automated, we observed that automation in changes is usually applied to patches and software installation, health checks, backups and configuration tasks. There is no attribute in the change ticket to indicate that the implementation is automated, but interviews with SMEs allowed us to infer automation from certain keywords in the change ticket description, for example, "apply fixes", "backup", and "hot fix". The accuracy of the keywords was checked against a random sample from each category. For categories for which the keywords proved a poor method of estimating full automation, a 2% sample of the category was manually categorized and that was used as an estimate for the category as a whole. Table V shows the categories for which automation is already applied or we considered that there is a high potential for full automation. For each of these categories, the "% Automation" column indicates the fraction of change tickets in that category that are either fully automated or have a high potential for full automation. Overall, we determined that 28% of all changes have potential for full automation. Change tickets that were not judged as either having a high potential for full automation or not automatable are considered "partially automatable".

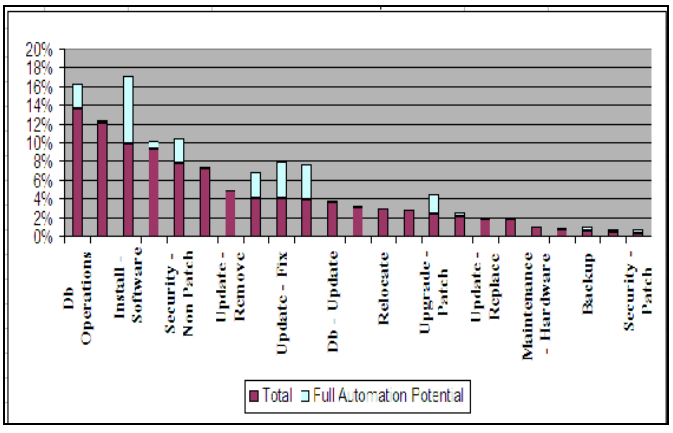

Fig.1. Full automation potential of the most common categories

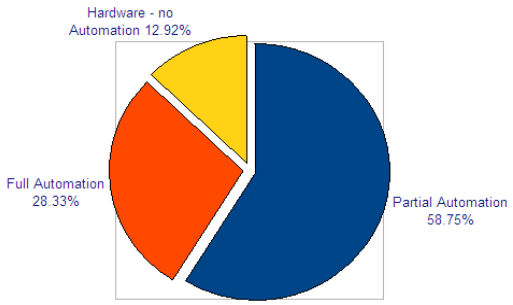

Fig 2. Automatable changes

Figure 1 compares the percentage of full automation with the total ticket percentage for the category. Overall, based on the analysis of change actions, we concluded that 58% of the changes across 13 categories were candidates for partial automation; while not fully automatable, these changes comprise a reasonable share of steps that can be automated while an SA has to intervene in the process and execute some steps manually. Thus, 58% of changes implementation plans can be reviewed and improved such that the likelihood of human error is reduced (Fig. 2).

IV. Human-Supervised Partial Automation

In this section, we propose a novel architecture and design principles for a tool workspace that facilitates the proactive prevention of service interruptions due to human errors. The proposal builds on the insights regarding the complexity of SA work presented by previous research [7-12] and our analysis of system interruptions, human error, and change activities in a large IT Service Support and Delivery environment in sections II and III.

The process implemented for partial automation is illustrated in Figure 3. It is based on the interactions of (1) a single, web-based interface for the SA into all service delivery processes (WISH), (2) data integration across all service delivery data, sources (ADI), (3) contextualization of the operations (COI) based on best practices and configuration information, (4) coordinated execution through dispatching (DI) and (5) controlled access to the targeted infrastructure (ACI).

An overall flow of the interactions comprises the following steps: A change request is entered into an incident, problem and change (IPC) system. This triggers an alert in the Dispatch system (DI). DI collects the necessary data though the Data Integration Adapter (ADI) and prepares the dispatch decision. When the request is dispatched, the assigned SME is notified. He or she retrieves the request details in the Web-based Interface (WISH) and accesses COI for customization based on existing artifacts for similar past changes. The newly created customized artifact is uploaded to COI and tagged with the change id. This triggers a request for dispatching of the change to an SA. The dispatcher assigns the change to an SA, who uses WISH to access the customized artifact associated with the change. When SA decides to implement the change, the WISH execution engine checks the access validity through ACI queries and triggers a log-in procedure into the targeted system.

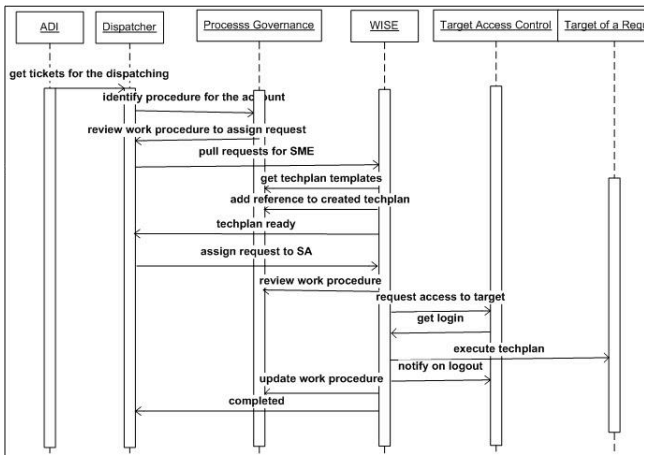

Fig. 3. Partial automation process sequence diagram

A. Adaptation and Integration of Data Sources

The Adapter component (ADI) integrates content across all of the data sources in IT Service Delivery, including the business model, IT Service Management (ITSM) systems, users, etc. While not directly responsible for human error prevention, ADI is the primary data input component and performs functions that are critical to all downstream human error prevention components. For the purposes of this paper, the business model consists of information about how services have to be delivered. It includes contractual agreements that define the relationship between the service provider and the customer, such as service level agreements (SLAs) and the catalog of exposed services, as well as business process information such as the catalog of internal services, technician roles, and internal policies.

A key function of the Adapter component is to combine the data from multiple heterogeneous IPC systems into a single data schema so that they can be easily consumed by downstream components. Where it is necessary to draw a distinction between the data actually in the IPC systems and their representation in the combined data schema, the former are named “tickets” and the latter “work orders”. A ticket includes information about:

- The type of incident or change as defined by a catalog of exposed services,
- Location information, including servers, networks, or databases to be affected by change or related to incidents,
- Reported severity, which together with SLAs affect the priority and scheduling of services, and
- Scheduling constraints beyond those specified in SLAs.

B. Coordinated execution through dispatching

The dispatcher is a person or group responsible for assigning work orders to SAs who then diagnose and correct the incident or problem, implement the change, or otherwise provide the service requested through the IPC. Most dispatchers work with a service delivery team composed of technicians with shared expertise in a single technical area, such as DB2 database or AIX operating system administration.

Each type of internal service is associated with one team, although a single team may be responsible for many service types. Where there is a direct relationship between an external service and a corresponding internal service, work orders can be routed automatically to the appropriate service delivery team and its dispatcher. More complex projects require division into multiple internal services and cooperation among multiple dispatchers associated with multiple service delivery teams to support these services. A dispatcher may break an internal service into one or more tasks depending on the business model and assign these tasks to one or more SAs. The tasks may explicitly have a required order of implementation (*e.g.*, configure the server *before* placing it on the network). Different technicians within a team may have different levels of training in different types of tasks. Assignment may require judgment weighing competing responsibilities, such as completing tasks efficiently vs. cross-training. The dispatcher annotates the work order with information about the schedules for time frames during which work is to be performed, as constrained by the technician assignments, ordering restrictions, SLAs, and customer scheduling constraints.

Dispatching is a complex process that can itself be a source of human error. Although mitigating this source of error is beyond the scope of the paper, there is reason to believe that partial automation can be applied to it as well. For example, an explicit model of technician skills, availability, and workload will make it possible to suggest appropriate technicians for each task while still giving the dispatcher room to make judgments involving higher-level issues such as balancing training requirements against efficiency.

C. Contextualization of Operations

An important measure in minimizing human error is to standardize the "best practice" process artifacts and provide delivery personnel ready access to these best practices. For a IT delivery organization, the complexity of the customers' IT environments and the dynamicity of the services required render the capture, standardization, and improvement of best practices a challenging task. In addition to capturing the best practices, making the knowledge base easily available is an important design requirement. Thus, there are two activities while a change is being prepared: help the user with references to existing changes and capture new changes and best practices from the current change being prepared.

IBM has developed a tool, called BPMS, which provides the capability to capture and improve the best practices. The goal of the BPMS is to provide the IT delivery community with a web based access to global repository of standard processes artifacts: processes, procedures, and work instructions including supporting documentation. BPMS employs social computing techniques to leverage the subject matter experts in the community to maintain the relevance and vitality of the content. Every artifact in BPMS has an owner who is responsible for the content and a group of reviewers and approvers. Owners are encouraged to leverage feedback from users for continually improving their artifacts. Process artifacts can be linked together and to various categories, such as the delivery catalog entry which applies the artifact and the accounts using the artifact. These links are used to navigate and filter artifacts to improve users' ability to locate artifacts of interest. Users can add to artifacts annotations which can be typed (*e.g.,* a variation) and restricted as applicable to an account or group of accounts. Through annotations users can register usage of an artifact, suggest improvements, and attach relevant information.

In order to help the user access the knowledge-base in BPMS, we have designed another tool, which acts like a recommender when the user prepares a change plan. As the user chooses a task, *e.g.*, apply a patch on AIX, the corresponding task stored in BPMS with matching keywords are displayed as recommendations. The user can use the recommendations to build the plan faster. Along with the recommendation feature, the user is also guided to follow the structure while preparing the work instructions. The introduction of a structure during work instruction preparation is essential in order to programmatically capture the insights of the changes. The change plan creation assist tool is currently Excel-based, and uses embedded macros to access the knowledge-base. The details of the tool are presented in [19].

D. Web-based interface for human – system interaction

The HEP framework provides the SAs with a web-based interface (WISH) for performing change operations in a partial automation approach. In a complete automation scenario, the change is triggered automatically. The process proceeds from the beginning to the end without human intervention. The complete automation may not be proper in many situations unless each step in the execution is guaranteed to be successful. The execution of the change contains a sequence of steps. The script for each step was written by human being therefore the script itself possibly contains errors. Even though the script proves to work well in one target, it could fail in a different target for the same change due to the different configuration or some other changes in the environment. Running the script step by step with human supervision (a partial automation) could improve the success rate. During the execution, the SA watches each step and is alerted only on exceptions or errors, The SA then makes a decision whether to continue or abort the execution to prevent worsening the situation. In a partial automation scenario, the SA interacts with the system through the Web GUI during the execution of change operations.

In this section we address this scenario of human-system interaction. The IPC system holds the change record describing the details of the work order. The request description includes the instructions for change as resulted from the contextualization of operations and the related servers and components subject to change. WISH collects all information related to the change request including the contextualized work instructions, assigned SA, and scheduled time window based on the work order and server or configuration item references. Because the scheduled start and end time are associated with each change request, the assigned SA can only start executing a ticket if it is within valid time window. The execution is triggered by accessing the work order using the web-based interface. During this

time, the interface presents concurrently the instructions to be executed and the execution context on the target server, which in the current implementation will run the UNIX operating system. The latter is presented in a customized "putty" window. If there are multiple servers to be accessed as part of the change, separate putty windows are invoked for logging into each server.

The user logs in through the putty windows to each server. Once the authentication succeeds, the SA requests (from the WISH dashboard) the execution of the next command. Upon SA action, the command is transferred to the putty window for execution on the target server. The GUI is a combination of JavaScript and applet technology. The JavaScript called as a result of pressing the "next command" button passes the command string to the putty window. The user can provide additional parameters to the command if required. For example, if the command issued is "kill" then the argument for this command is the process identifier obtained manually from the output the previous command "ps".

Another component of the command execution tool is a server-side component responsible for maintaining the state of command execution. The component is implemented as an application running on portal software. All the commands executed by the SA as well as the output of the commands are automatically logged. After the work order execution completes, the audit logs are uploaded to a database for later analysis.

When a work order execution has to be aborted *(e.g.*, because of incomplete command or change of the plan), the changed description and the status of the execution (*e.g.*, next step) are stored for future reference using an upload interface. As a result, the work order state is switched back to executable, allowing the SA to resume the execution. The changes to the work order performed during change implementations are applied by an SME that is identified in the change ticket. All plan change actions are audited in order to assess if an error occurred and what caused it. Further details of the system can be found in [18].

In the general, the design of the web-based interface WISH, integrating across services and content, follows the principles developed by HCI experts [17] in order to prevent errors. For instance, the GUI is the first gate to block human errors into the IT system. Also, the GUI offers help messages and warnings in order to alert the user on the risks of his actions. However, the distinction between necessary and excessive help should be considered. Good feedback with suggested actions for error recovery or even error prevention highly benefits the overall user experience.

E. Controlled Access

The goal of the *Access Control* component (ACI) is to prevent 'wrong target' errors. The control of access to a targeted system is directly taken from the human user and placed on the component, which maintains information on the 'eligibility' of a given user to access a particular target. In our implementation, the SA has access to a system only in the context of a specific operation during the scheduled time for the operation. For example, typically, an SA has read-only access to a system that he is responsible for maintaining. His or her permissions change to include write and execute when there is an active request related to the system he or she is maintaining. ACI queries the data sources through the Adapter periodically to get current information on the necessary changes to access level. Collectively with *WISH,* which monitors user actions, ACI also addresses 'wrong time' errors by revoking user's access to the target when the time-window is closed.

V. Related Work

A. Human Factors Research

With the goal of improving operational performance and safety, Human Factors research tries to understand why and how accidents or errors occur and how they can be prevented. Many models for characterization of human errors have been developed, most of them focusing on the conditions that triggered the error. Such models relate to (1) individual, task, or organization, (2) individual vs. system [2], or (3) insufficiency of knowledge, skills, or rules [6]. Typical prevention is based on training, well designed interfaces, detection and recovery from procedure failures, and automation [1]. In our work, we depart from traditional Human Factors research and consider a functional characterization of human error into the 4 Wrongs, which identify actionable items that drive our solutions for human error prevention.

B. IT Service Delivery: Work Environment and Tools

A significant body of previous research has focused on the work of system administrators and the effectiveness of their tools with respect to productivity and risk reduction. While not specifically interested in preventing human error, many of the proposed tool design principles contribute to this end.

The studies in [7,8,10,11,12] are based on a series of field studies in large corporate data centers that reveal the challenging, human error-prone work environment. The contributing factors [11] include multitasking across interleaved and parallel workflows, tool command languages that cannot prevent small types from triggering high-risk procedures, diversions caused by problems that arise outside the scope of the initial problems, and the multitude of tools that have to be used by SAs to accomplish their tasks. The methods used to prevent errors include planning and rehearsals, customization and automation with tools they build and trust.

Haber [10] considers 'sensemaking' in the context of SA activities as the integration of information about all of the components relevant for the task-at-hand. [7,10] identify the large volume of information and the distribution across multiple sources, people, or databases. [7] proposes to address this complexity with a distributed cognition approach focused on (1) the information flow among participants in the process and sources of information and (2) how the information flow affects how information is received and used.

[8] highlights the disconnect between SA tools and the complexity of the managed systems and of the tasks performed, the risk and stress derived mission critical applications, the ongoing development of script and tools. Recommendations to improve the tools used by SAs span from standardization of configuration and terminology, tool integration, support for validation of configurations, easy to use interfaces, support for situational awareness, online and

offline collaboration. While [8] is focused on tools to manage individual system components like operating system, web applications or databases, we submit that similar principles proposed should be expanded to frameworks for implementation of IT management processes, such as the HEP Framework, in order to achieve low human-error levels.

[12] addresses the challenges of managing the complex IT Service Delivery environments through automation. Field studies reveal that automation tools need to support (1) rehearsal and planning, (2) maintaining situation awareness, and (3) managing multitasking, interruptions and diversion. For instance, automated tools should allow building test systems or to quickly undo changes. The study observes that automation is likely to reduce situation awareness and propose a visualization approach that mitigates this limitation. Finally, automation increases the likelihood of multitasking and diversions, thus increases the risk error. The limited applicability of automation to rather simple tasks and the negative impact on 'system visibility' is also addressed in [4]. Our analysis concurs with this view, showing that automation is not likely applicable to a large set of change operations in an IT Delivery organization. We propose the use of partial automation as a way to expand the human-error reduction beyond the limitations of automation. With the SA guiding the execution of automated steps, partial automation in change implementation is less prone to have negative effects due to the mismatch of mental and computer models, such as in human-computer cooperative problem solving [4].

Previous work proposed a variety of tools to support SAs in management of personal activity [13], of structured and unstructured work activities [14] such as change management, of change scheduling decisions [15, 16], and of automated the software installation process [17]. The HEP framework proposed in this paper fosters tool integration on the basis of foundational elements that prevent the 4 Wrongs.

VI. Conclusion

We have presented an analysis of a large volume of change tickets. Using our categorization of change operations with respect to action and operation type, we have gained insights into how human errors occur and how these errors could be reduced through partial or complete automation. We believe the lessons drawn from this analysis have general application in change implementation and the analysis itself has general application to other problems in IT Service Support and Delivery.

We have proposed a "Wrong-driven" framework, the HEP Framework, for reducing human error in the implementation of resolutions for IT change requests. This framework proposes specific tool design approaches to reduce the occurrence of each of the "4 Wrongs":

- catalog-based specification of change requests parameterized by organization or workgroup to address "wrong request" errors,
- control of SA access by CI and change window to address "wrong CI" and "wrong time" errors,
- partial automation of change implementation scripts to address "wrong command" errors.

In future work, we expect to test our framework in the field to determine whether it does reduce the "4 Wrongs" as predicted by our analysis in section IV.

References

[1] A. B. Brown. "Oops! Coping with Human Error in IT Systems," *Queue*. vol. 2, no. 8, Dec. 2004.

[2] J. Reason. "Human Error: Models and Management," BMJ, 320: 768-770, 2000.

[3] D. Patterson et al. *Recovery Oriented Computing (ROC): Motivation, Definition, Techniques and Case Studies*, Computer Science Technical Report UCB//CSD-02-1175, U.C. Berkeley, 2002.

[4] S. A. Guerlain, P. J. Smith, S. M. Gross, T. E. Miller, J. W. Smith, J. R. Svirbely, S. Rudman, *Critiquing vs. partial automation: How the role of the computer affects human-computer cooperative problem solving.* Human performance in automated systems: Current research and trends, pp. 73-80. Lawrence Erlbaum Associates, Mahwah, NJ, 1994.

[5] S. Dekker. *Ten Questions about Human Error: A New View of Human Factors and System Safety*. Mahwah, NJ: 2005.

[6] J. Reason. *Human Error,* Cambridge University Press, 1990.

[7] P. Maglio, E. Kandogan, E. Haber. *Distributed Cognition and Joint Activity in Computer-System Administration*. Springer, New York, NY, 2004

[8] E. Haber, J. Bailey. *Design guidelines for system administrator tools developed through ethnographic field studies* Proceedings of the Computer-Human Interaction for the Management of Information Technology (CHIMIT '07). Cambridge, MA, USA, 2007

[9] ITIL. Available: http://www.itil-officialsite.com/home/home.asp

[10] E. Haber. *Sensemaking Sysadmins: Lessons from the Field,* Sensemaking Workshop at ACM CHI, 2005.

[11] R. Barrett, E. Kandogan, P. Maglio, E. Haber, L. A. Takayama, M. Prabaker. *Field Studies of Computer System Administrators: Analysis of System Management Tools and Practices* CSCW'04.

[12] R. Barret, P. Maglio, E. Kandogan, J. Bailey. *Usable Autonomic Computing Systems: the Administrator's Perspective* Autonomic Computing 2004.

[13] V. M. Gonzalez, L Galicia, J Favela. *Understanding and supporting personal activity management by IT service workers* Proceedings of the 2nd ACM Symposium on Computer Human Interaction for Management of Information Technology (CHIMIT '08).

[14] J. Bailey, E. Kandogan, E. Haber, and P. Maglio. *Activity-Based Management of IT Service Delivery,* CHIMIT 2007.

[15] :J. Sauve, R. Reboucas, A. Moura, C. Bartolini, A. Boulmakoul, and D. Trastour. *Business-driven support for change management: Planning and scheduling of changes*, Proceedings of IFIP/IEEE International Workshop on Distributed Systems: Operations and Management, 2006.

[16] L. Zia, Y. Diao, D. Rosu, C. Ward, and K. Bhattacharya. *Optimizing Change Request Scheduling in IT Service Management.* SCC2008

[17] A. Keller, J. Hellerstein, J. Wolf, K. Wu, and V. Krishnan. "*The CHAMPS system: Change management with planning and scheduling,*" in Proceedings of IFIP/IEEE Network Operations and Management Symposium, 2004.

[18] J. Nielsen, H. Loranger. *Prioritizing Web Usability.* New Riders Press, Berkeley CA, 2006.

[19] Venkat Madduri, Manish Gupta, Pradipta De, Vishal Anand. "Towards Mitigating Human Errors in IT Change Management Process," Proceedings of the ICSOC, 2010.

[20] L. Shwartz, N. Ayachitula, M. Buco, G. Grabarnik, S. Maheswaran, C. Ward, S. Weinberger, "IT Service Provider's Multi-Customer and Multi-Tenant Environments," E-Commerce Technology, IEEE International Conference on, and Enterprise Computing, E-Commerce, and E-Services, IEEE International Conference on, pp. 559-566, CEC-EEE, 2007