Story Version Control and Graphical Visualization for Collaborative Story Authoring

Fabio Zünd
ETH Zurich
fabio.zund@inf.ethz.ch

Steven Poulakos
Disney Research Zurich
steven.poulakos@disneyresearch.com

Mubbasir Kapadia
Rutgers University
mubbasir.kapadia@rutgers.edu

Robert W. Sumner
ETH Zurich/Disney Research Zurich
robert.sumner@inf.ethz.ch

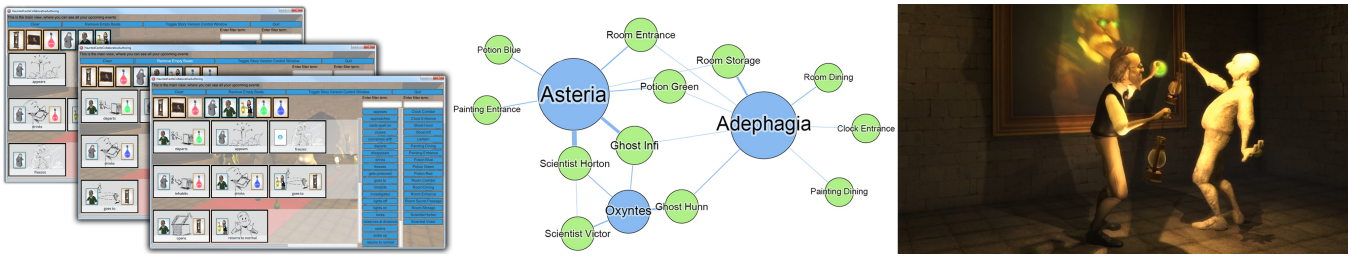


Figure 1: Collaborative story authoring in the visual editor, graph-based story visualization, and playing back an animated story.

ABSTRACT

This paper presents a story version control and graphical visualization framework to enhance collaborative story authoring. We propose a media-agnostic story representation based on story beats, events, and participants that describes the flow of events in a storyline. We develop tree edit distance operations for this representation and use them to build the core features for story version control, including visual diff, conflict detection, and conflict resolution using three-way merge. Our system allows authors to work independently on the same story while providing the ability to automatically synchronize their efforts and resolve conflicts that may arise. We further enhance the collaborative authoring process using visualizations derived from the version control database that visually encode relationships between authors, characters, and story elements, during the evolution of the narrative. We demonstrate the efficacy of our system by integrating it within an existing visual storyboarding tool for authoring animated stories, and additionally use it to collaboratively author stories using video and images. We evaluate the usability of our system with two user studies. Our results reveal that untrained users are able to use and benefit from our system. Additionally, users are able to correctly interpret the graphical visualizations and perceive them to benefit collaboration during the story authoring process.

CVMP 2017, December 11–13, 2017, London, United Kingdom

ACM ISBN 978-1-4503-5329-8/17/12…$15.00
https://doi.org/10.1145/3150165.3150175

CCS CONCEPTS

• **Human-centered computing** → **Visualization systems and tools**;
• **Computing methodologies** → **Graphics systems and interfaces**;

KEYWORDS

story authoring, collaboration, visualization, story version control

ACM Reference Format:
Fabio Zünd, Steven Poulakos, Mubbasir Kapadia, and Robert W. Sumner. 2017. Story Version Control and Graphical Visualization for Collaborative Story Authoring. In *CVMP 2017: 14th European Conference on Visual Media Production (CVMP 2017), December 11–13, 2017, London, United Kingdom.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3150165.3150175

1 INTRODUCTION

Stories pervade our daily lives. They document our histories, they educate us, they entertain us, and they inspire us. The maturity in connected platforms for content authoring and sharing has led to a paradigm shift where traditional consumers are now active content creators. In this connected setting, collaboration naturally plays a central role in the narrative authoring process for media productions. However, current authoring tools are not explicitly designed for collaborative authoring and suffer from two primary limitations. First, without native support for collaboration, authors must explicitly communicate with one another to manually synchronize their content and resolve conflicts. Second, understanding the overall progression of contributions and the relationships to creative content in a collaborative setting with multiple authors is nearly impossible due to inherent complexities in such a setup. As a result, patterns in authoring procedures that might enhance collaborative creativity are difficult to find.

Our research enhances collaborative story authoring by providing a Story Version Control (SVC) and graphical visualization framework (Figure 1). Our system allows authors to work independently on the same story while providing the ability to automatically synchronize their efforts and resolve conflicts that may arise. Additionally, we provide visualization tools that extract and visualize information from the version control database to convey the creative intent of authors, their contributions to the story over the course of its evolution, and their relationships with characters and other authors. Together, these capabilities enhance collaborative authoring and provide meaningful insights into the creative process.

We address several challenges in developing our version control and visualization framework. First, we present a tree-based, media-agnostic story representation based on story beats, events, and participants that describes the flow of events in a storyline. Given this representation, we develop tree edit distance operations to compute the distance between two stories as well as edit operations needed to transform one story to another. With this foundation, we build the core features for SVC, including visual diff tools, conflict detection, and conflict resolution using three-way merge operations.

In addition to developing the base functionality for SVC, we further demonstrate that the data within the story-oriented version control database holds great potential to enhance the collaborative authoring process through visual representations of story evolution. We propose three types of visualizations that span story content and author participation. First, story-centric visualizations represent either character or author involvement in story progression. Second, relationship visualizations depict the connectivity between authors, characters, and story elements. For example, co-authors may observe the characters an author most frequently edits. Finally, meta-relationship visualizations characterize the correlation between authors based on their interactions with story characters or story events. Two authors whose edits frequently include the same character may have a strong meta-relationship.

We demonstrate our Version Control System (VCS) and information visualization tools on collaboratively authored stories using an animation synthesis engine. To show the media-agnostic nature of our method, we further show stories based on raw images and video. We validate the efficacy of our tools with a user study in which teams of authors collaboratively created novel stories. Additionally, users are able to correctly interpret visualizations to extract meaningful information and perceive it to be a useful tool to promote collaboration in storytelling.

2 RELATED WORK

Story Authoring. From cave paintings to modern day storyboards [5] and comics [10], telling stories with pictures has evolved to be more expressive and dynamic. Digital Storytelling has come to represent a variety of multimedia formats (e.g. video, still images, animation and audio) that enable people to share personalized stories. The flexibility offered by digital formats, makes it easier to experiment with the interaction between sequences of visual information, deriving new meaning for individual multimedia elements based on montage.

Technology also evolves to support story creation. Emergent forms of storytelling are enabled by technologies that guide the temporal ordering of shots, sequences, and scenes based on attributes including characters, emotions, themes, and story structure [14]. Artificial intelligence, in the form of automated planners, can also support consideration of cinematic actions and scene composition of those actions [7] as well as maintaining consistency between shots and enable shot reuse [12].

Interactive narrative systems, which allow users to interact with the story world and hence become part of the narrative, are growing into applications in education, training, and entertainment [13]. Visual authoring tools such as CANVAS [8] leverage computer-assistance to visually author and synthesize multi-character animations from sparsely specified narrative events. Our system enables authors to specify a sequence of narratively significant elements of multimedia content to generate stories in a media-agnostic way.

The continued evolution of web technologies has enabled more participatory and collaborative forms of content production. Early observations of collaborative writing in scientific communities have observed that the process of collaborative writing is a dynamic process with continuous negotiations related to both the written content as well as roles and responsibilities between the co-authors [2]. Our system aims to support the collaborative process by facilitating awareness of story contents and author participation.

Version Control Systems. A VCS aims at maintaining a revision history and at facilitating and supporting collaboration between users. We refer the reader to the survey paper on model versioning approaches [1], which describes various version control models and approaches in detail. Like modern VCS used for code versioning, such as SVN[1] and Git[2], SVC implements an optimistic control mechanism with three-way merging.

VCS for other domains have been developed in recent years. MeshGit [4] and skWiki [16] are prime examples for non-traditional systems. They implement version control for editing meshes and for editing multimedia projects, respectively. At the core, they define an edit distance in that particular domain and provide version control functionality such as, checkout, commit, and update to the user, that operates on the repository based on an edit distance. MeshGit defines the mesh edit distance as a measure of dissimilarity between meshes, which can be used to transform a mesh into another mesh. skWiki expresses the edit distance between two media as a set of transformation operations, represented in domain specific language. We introduce a VCS in the domain of stories. In contrast to MeshGit and skWiki, we define an abstract domain representation for stories and remain media-agnostic such that any form of story (e.g. based on animation, video, or comics) is compatible with the system.

Information Visualization. Humans perceive visual attributes very well, which motivates the mapping of different data to visual attributes such as color, size, and proximity [9]. Information visualization is a cognitive activity facilitated by visual representations, which make it possible to explore relationships, to confirm hypothesis on data relationships, and to aid the construction of cognitive models [9]. Our aim is to leverage our data representations and customize the visualization capabilities of existing tools to provide intuitive representations of information about the collaborative authoring process.

Contributions. Our work complements existing work in digital story authoring and computational narrative by providing the tools

[1] https://subversion.apache.org/ [2] https://git-scm.com/

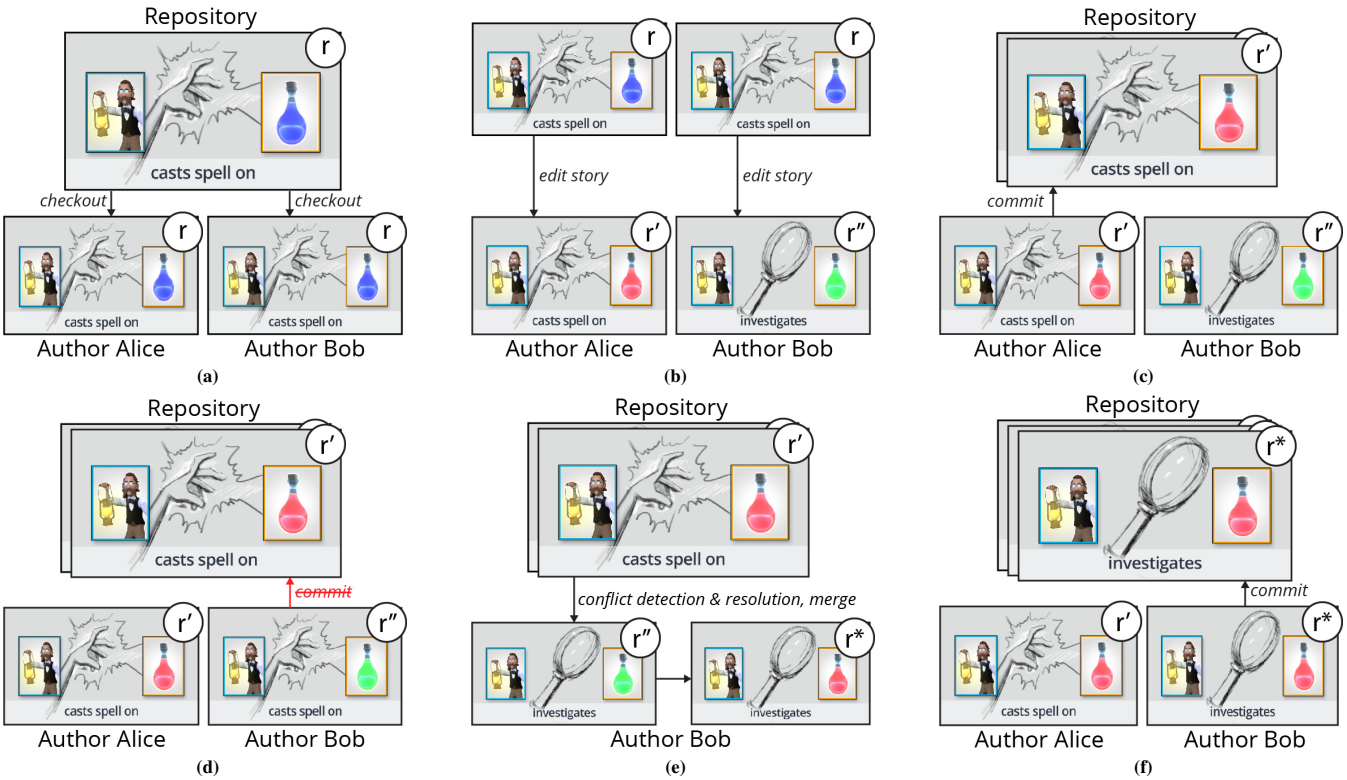


Figure 2: Example scenario illustrating SVC's workflow utilizing an optimistic versioning approach.

for story authors to seamlessly and efficiently collaborate. This is accomplished using two important novel contributions: (1) A VCS for stories that relies on a media-agnostic representation of stories, and (2) Information visualization tools to help author glean meaningful information from multi-user story authoring sessions, in an effort to understand how the content of a story evolves and how authors collaborate.

3 STORY REPRESENTATION

SVC assumes an abstract, tree-based, media-agnostic representation of a story. Inspired by CANVAS [8], we define the domain knowledge of a story as the set of story building blocks an author employs to create a narrative. These building blocks are story beats, story events, and event participants. A participant is a character or prop object that takes part in the story in one or multiple events. An event is a context-specific interactions between any number of participants, where each instance of an event can have a different outcome depending on the participants. A beat combines multiple simultaneously happening events into one unit of advancement in story time. SVC implicitly assumes progression of time in the story as progression of beats. Events within a beat are unsorted and happen in parallel.

A story $\mathbf{s}$ is represented as an ordered tree in which the story node $\mathbf{s} = \langle I, \{\mathbf{b}_0, \mathbf{b}_1, \ldots\}\rangle$ is the root node containing beat nodes $\mathbf{b}$ as ordered children. A node ID I is contained in every node. A beat node $\mathbf{b} = \langle I, \{\mathbf{e}_0, \mathbf{e}_1, \ldots\}\rangle$ contains an ordered list of event nodes. An event node $\mathbf{e} = \langle I, g, \{\mathbf{p}_0, \mathbf{p}_1, \ldots\}\rangle$ contains a signature name g, which defines the type of event, and participant nodes $\mathbf{p}$. The participant nodes $\mathbf{p} = \langle I, h\rangle$ are leaf nodes, they contain a participant ID h. Figure 3 illustrates the abstract tree structure of a story.

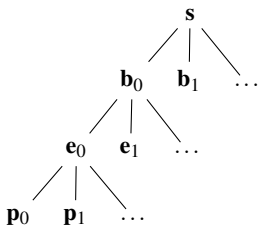


Figure 3: Tree representation of a story using story nodes s, beat nodes b, event nodes e and participant nodes p.

We define a path $\mathbf{P_n} = \{I_0, I_1, \ldots, I_k\}$ of a node $\mathbf{n} \in \{\mathbf{s}, \mathbf{b}, \mathbf{e}, \mathbf{p}\}$ as the concatenation of node IDs I from the root node $\mathbf{s}$ at level 0 to node $\mathbf{n}$ at level k. The notation $\mathbf{n} \succ \mathbf{n}'$ expresses that a node $\mathbf{n}$ is a descendant and lies in the sub-tree of $\mathbf{n}'$. The node $\mathbf{n}$ is a descendant of $\mathbf{n}'$ if its path is longer $|\mathbf{P_n}| > |\mathbf{P_{n'}}|$. Finally, we use the symbol $\mathbb{S}$ to denote the space of all possible stories. By using this tree representation, we benefit from existing tree editing algorithms to compare, modify, and merge stories.

Authoring a story requires two main tasks. First, the domain knowledge of the story is defined by an expert. The expert defines

events and participants that exist in the story world. Second, the author creates instances of events, orders them into beats, and assigns participants to events, in a storyboarding fashion. A screenshot of the authoring editor is depicted in Figure 6a.

4 STORY VERSION CONTROL

SVC utilizes the optimistic versioning paradigm [1] in which a three-way merging is applied. Three-way merging is common in most of today's VCS and allows the system to identify editing operations of authors more precisely than using raw-merge and two-way merge.

We illustrate the three-way merging authoring paradigm by an example scenario in Figure 2 in which authors Alice and Bob are collaboratively editing a story about a character named Horton. (a) The authors checkout the latest revision $\mathbf{r}$ from the repository, which will serve as their base revision. The current revision $\mathbf{r}$ contains the story *Horton casts a spell on a blue potion*. (b) The authors each make changes to the story, creating new revisions $\mathbf{r}'$ and $\mathbf{r}''$, respectively. Alice changes the *Potion Blue* to a *Potion Red*. Bob changes the *Potion Blue* to a *Potion Green* and the event from *casts a spell* to *investigates*. These changes are partially overlapping (changing the potions) and thus conflicting. (c) Alice commits her changes $\mathbf{r}'$ back to the repository. (d) Bob cannot commit his changes $\mathbf{r}''$ to the repository in the current state, as his base revision $\mathbf{r}$ is older than the current revision $\mathbf{r}'$ on the server. (e) Instead, he is required to first update and merge his story $\mathbf{r}''$ with the latest revision $\mathbf{r}'$. As a conflict is present, SVC assists Bob in resolving the conflict and merging his revision with the latest, thereby creating revision $\mathbf{r}^*$. (f) Bob can now commit his updated revision $\mathbf{r}^*$ to the repository. Afterward, Alice can update her revision $\mathbf{r}'$ conflict-free to the latest revision $\mathbf{r}^*$.

4.1 Story Edit Distance

In SVC, version control functionality such as checkout, commit, update, and merge, is provided through tree edit operations. Tree edit operations are employed as a representation of dissimilarity between two stories and thus constitute the core of the repository. The following sections describe how tree edit operations are applied to support version control mechanisms.

Using the Robust Tree Edit Distance (RTED) [11], SVC calculates an injective mapping $\sigma : \mathbb{N}^+ \to \mathbb{N}^+$ that maps post-order node indices in a story $\mathbf{s} \in \mathbb{S}$ to post-order node indices in another story $\mathbf{s}' \in \mathbb{S}$. In the example story trees depicted in Figure 4, the mapping is $\sigma = \{1 \mapsto 1, 3 \mapsto 2, 4 \mapsto 4\}$.

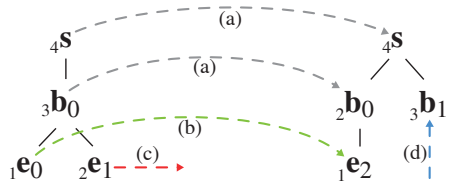


Figure 4: Example mapping of two story trees to illustrate (a) ignored mappings, (b) rename operations, (c) remove operations, and (d) insert operations.

Given a mapping, the corresponding tree operations can be synthesized as follows:

- **(a)** Mappings between equal nodes are ignored.
- **(b)** For each mapping between a node $\mathbf{n} \in \mathbf{s}$ and $\mathbf{n}' \in \mathbf{s}'$ a **rename operation** is synthesized if the nodes are not equal, that is, if their ID is not identical $I_{\mathbf{n}} \neq I_{\mathbf{n}'}$. A rename operation $o^{\text{ren}}\mathbf{s}, \mathbf{P_n}, \mathbf{n}' : \mathbb{S} \to \mathbb{S}$ updates the node $\mathbf{n} \in \mathbf{s}$ at path $\mathbf{P_n}$ to a node $\mathbf{n}'$.
- **(c)** For each node in $\mathbf{s}$ that is not mapped to a node in $\mathbf{s}'$, a **remove operation** is synthesized. A remove operation $o^{\text{rem}}\mathbf{s}, \mathbf{P_n} : \mathbb{S} \to \mathbb{S}$ removes the node $\mathbf{n} \in \mathbf{s}$ at path $\mathbf{P_n}$.
- **(d)** For each node in $\mathbf{s}'$ that has no mapping from a node in $\mathbf{s}$, an **insert operation** is synthesized. An insert operation $o^{\text{ins}}\mathbf{s}, \mathbf{P_n}, c, \mathbf{n}' : \mathbb{S} \to \mathbb{S}$ inserts a child node $\mathbf{n}'$ to the parent node $\mathbf{n} \in \mathbf{s}$ at path $\mathbf{P_n}$ at child index c.

These three types of operations are sufficient to transform an arbitrary story $\mathbf{s}$ into a new story $\mathbf{s}'$ [15]. The difference between two stories $\mathit{diff}(\mathbf{s}, \mathbf{s}') = \{o_0, o_1, \ldots, o_n\}$ is a set of n operations o_i that express the atomic changes required to transform $\mathbf{s}$ into $\mathbf{s}'$.

4.2 Repository representation

The SVC repository stores and keeps track of all operations the authors have performed over time. We define a story repository $\mathbf{R} = \{\mathbf{r}_0, ..., \mathbf{r}_N\}$ as an ordered list of revisions. Inspired by [16], a revision $\mathbf{r}_i = \langle u_i, a_i, b_i, t_i, \mathit{diff}_i \rangle$ is a tuple consisting of a unique identifier u, an author a, a reference to a parent (base) revision b, a time stamp t, and a diff *diff*. The repository does not store snapshots of a story but instead accumulates the delta transformation operations. Applying the tree edit operations in a diff diff_i to the story $\mathbf{s}_{i-1}$ from the previous revision $\mathbf{r}_{i-1}$ recreates the current story $\mathbf{s}_i$. Hence, recreating a specific story $\mathbf{s}_k$ can be expressed as a function composition series $\mathbf{s}_k = \mathit{diff}_k \circ \mathit{diff}_{k-1} \circ \ldots \circ \mathit{diff}_0 \mathbf{s}_\emptyset$.

4.3 Version Control Conflicts

In a three-way merge scenario, two users try to commit their story to the repository, both stories based on the same base revision story, as illustrated in Figure 2. Two diffs need to be merged, that is, the two sets of operations need to be combined. During that process, SVC detects conflicting operations and lets the authors resolve the conflict manually using the client. Two operations are conflicting under the following conditions:

Rename Conflict A rename operation $o_i^{\text{ren}}\mathbf{s}, \mathbf{P_n}, \mathbf{n}'$ conflicts with a rename operation $o_j^{\text{ren}}\mathbf{s}, \mathbf{P_n}, \mathbf{n}''$ if both operations update the same node $\mathbf{n} \in \mathbf{s}$ to a different node, $\mathbf{n}' \neq \mathbf{n}''$.

Insert Conflict An insert operation $o_i^{\text{ins}}\mathbf{s}, \mathbf{P_n}, c, \mathbf{n}'$ conflicts with an insert operation $o_j^{\text{ins}}\mathbf{s}, \mathbf{P_n}, c, \mathbf{n}''$ if both operations insert a different node $\mathbf{n}' \neq \mathbf{n}''$ at the same child index c at the same parent node $\mathbf{n}$.

Remove Conflict A remove operation $o_i^{\text{rem}}\mathbf{s}, \mathbf{P_n}$ conflicts with any rename operation $o_j^{\text{ren}}\mathbf{s}, \mathbf{P_{n'}}, \mathbf{n}^*$ or insert operation $o_j^{\text{ins}}\mathbf{s}, \mathbf{P_{n'}}, c, \mathbf{n}^{**}$ if $\mathbf{n}' \succ \mathbf{n}$ holds, that is, if the latter operations target a descendant node of $\mathbf{n}$.

In the example illustrated in Figure 2 (c), Alice's and Bob's revisions $\mathbf{r}'$ and $\mathbf{r}''$ contain operations that conflict with each other. Alice's revision contains a rename operation changing the *Potion*

blue to the *Potion Red* and Bob's revision contains a rename operation changing the same *Potion Blue* to a *Potion Green* instead. The other operation in Bob's revision $\mathbf{r}''$ is a rename operation changing the event *casts spell on* to *investigates*, which does not conflict with any of Alice's operations.

4.4 System Architecture

Figure 5 summarizes the proposed system architecture. Story repositories are centralized and persisted on a server and can be accessed online by both story editing as well as story visualization clients.

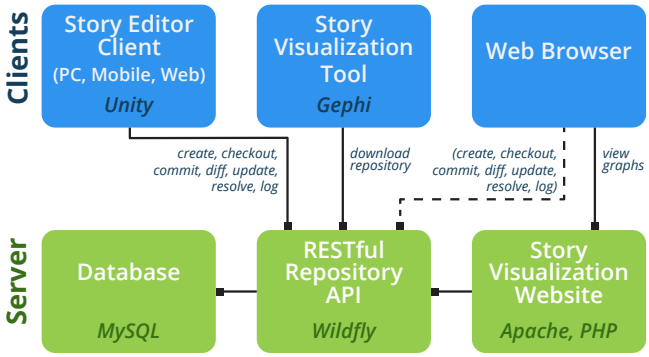


Figure 5: SVC System Architecture. Story repositories are centralized and persisted on a server and can be accessed online by both story editing as well as story visualization clients.

SVC Server. The SVC server, running on Ubuntu in the Amazon Web Services cloud[3], comprises a Wildfly[4] application that provides a RESTful API for performing operations on the repository and transmits json-serialized repository data. All data is persisted in a local MySQL database[5]. The Wildfly server application provides the following JAX-RS web services: *checkout, commit, diff, resolve, update, log, full, create*.

The **full** request is only used by the visualization tools to download the entire repository including the operations in all revisions as well as a story snapshots for each revision. Using this information, the tools have complete freedom to visualize any aspect of the repository.

During a **checkout** request for a revision $\mathbf{r}_k$, all sets of operations (diffs) are applied to the empty story in order to synthesize the requested story $\mathbf{s}_k = diff_k \circ diff_{k-1} \circ \ldots \circ diff_0 \mathbf{s}_\emptyset$.

A **diff** request calculates and returns the diff of two stories. The stories can be submitted or referred to using revision IDs.

During a **commit** request, a diff between the submitted story and the story is calculated. The diff is then appended to the repository as a new revision.

An **update** request from the client contains the author's latest story $\mathbf{s}_{\text{author}}$ as well as the author's base revision $\mathbf{r}_{\text{base}}$ with story $\mathbf{s}_{\text{base}}$. Using the story $\mathbf{s}_{\text{head}}$ in the latest (HEAD) revision in the repository the server calculates the diffs $diff\mathbf{s}_{\text{head}}, \mathbf{s}_{\text{base}}$ as well as $diff\mathbf{s}_{\text{author}}, \mathbf{s}_{\text{base}}$ and calculates the conflicts thereof. If there are no conflicts, the operations in $diff\mathbf{s}_{\text{head}}, \mathbf{s}_{\text{base}}$ are applied to $\mathbf{s}_{\text{author}}$ and the resulting story is returned to the client. If conflicts are present, they are sent to the client. The client presents the conflicts to the author in a visual fashion and, for each pair of conflicting operations, lets the author resolve each conflict by deciding which author's operations should be used. It is possible that operations from the same author are dependent from each other. A dependency exists between an operation $o(\mathbf{n}, \cdot)$ on node $\mathbf{n}$ and another operation $o'(\mathbf{n}', \cdot)$ on node $\mathbf{n}'$ if $o(\mathbf{n}, \cdot)$ is a rename or an insert operation and $\mathbf{n}' \succ \mathbf{n}$ holds. After each resolved conflict, the client applies the same decision for all dependent remaining conflicts. For instance, author Alice inserts a new event and a participant into a beat. These operations conflict with Bob's operation to remove the entire beat. Alice is resolving the conflicts. Her operation that inserts the participant depends on the operation that inserts the event and thus cannot be chosen independently. If Alice decides to keep her operation that inserts the participant, the client would automatically decide to keep as well the operation that inserts the event. After all conflicts have been resolved, the client sends a **resolve** request back to the server containing a decision for each conflict. The server then applies each selected operation in the conflicting operations pairs as well as the remaining non-conflicting operations in $diff\mathbf{s}_{\text{head}}, \mathbf{s}_{\text{base}}$ and $diff\mathbf{s}_{\text{author}}, \mathbf{s}_{\text{base}}$ to the base story $\mathbf{s}_{\text{base}}$ and sends the resulting story back to the client.

Story Editor Client. The Unity-based[6] story editor client application extends an existing animated story authoring framework [8]. It provides an intuitive interface and employs a storyboarding paradigm for authoring stories. Storyboard images serve as event objects and portrait images serve as participant objects. The story is authored by arranging participants, events, and beats in the 2-D screen space. The authoring editor is extended with a UI for performing version control tasks and visualizing story differences. After an update as well as when diffing two revisions, changed events and participants are highlighted using different colors. Figure 6a depicts a screenshot of the editor while an author is working on a story.

Story Visualization Tools. The Gephi[7] visualization platform was extended to connect to the SVC server's API and load the repository data structure. Using the **full** request, an entire repository including all operations and a story snapshot for each revision is constructed on the server and downloaded. The user can trim the repository to visualize only a specific interval of revisions. Gephi is specialized in graph-based visualizations and provides the user with a variety of statistical metrics and filters. A screenshot of the standalone graph visualization tool is depicted in Figure 6b. Various types of graphs, as described in Section 5, can be generated and explored by the user.

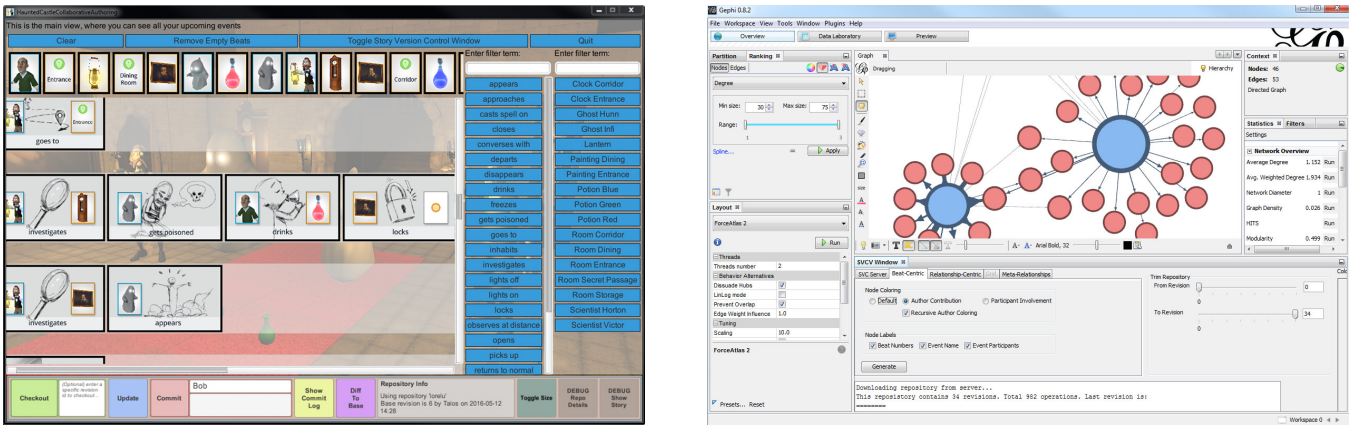
Additionally, we provide a website, which lets the author generate a predefined subset of graphs offered in the standalone visualization tool. This website targets users who decide to relinquish the features offered by Gephi but instead prefer a faster and simpler solution to explore graph visualizations of their stories.

5 GRAPHICAL VISUALIZATION FOR COLLABORATIVE STORY AUTHORING

Collaboratively authored stories serve as a multi-dimensional space of interconnected information. When we author stories collaboratively, it is valuable to explore the stories through different lenses. In this section we explore such different perspectives. Our dimensions are authors, revisions, participants, beats, and events. Each slice of

[3] https://aws.amazon.com/ [4] http://wildfly.org/ [5] http://www.mysql.com/

[6] https://unity3d.com/ [7] https://gephi.org/



(a) (b)

Figure 6: Story Version Control client applications. (a) The SVC Story Editor Client providing an intuitive drag-and-drop interface for authoring stories by arranging beats, events, and participants. A version control UI is integrated. (b) The Gephi-based story visualization tool with integrated SVC connectivity.

information characterizes valuable insights into the evolution of the story, characters, authors, and their relations, and is a precursor to facilitating collaboration.

Graphs offer intuitive visual metaphors to convey this information, where each of the dimensions can be assigned a visual attribute, such as node color, node size, or edge width. Different graphs can be constructed by varying visual attributes and aggregating dimensions. To illustrate different types of graphs, Figure 7 (left) depicts a simple repository containing three revisions from three authors, Alice, Bob, and Charlie. Purple highlighting indicates inserted events and participants while orange highlighting indicates changed events and participants since the last revision. Remove operations are not visualized. We refer to this exemplary repository in the following sections while introducing the graphs supported by SVC.

Story-Centric Graph. In a story-centric graph, beat and event nodes form the graph layout, as depicted in Figure 7 (a). Author contribution is aggregated over a revision interval and displayed inside the nodes as pie charts. Contributions are calculated recursively to include edits to events and participants within the beat. The story at the last value of the revision interval is visible. This graph visualizes the length, number of beats, as well as the width, number of parallel events, of a story. Figure 7 (a) illustrates that, while Alice created the first beat, she has only little contribution because multiple operations were applied to it by Bob afterward. In contrast, Charlie is the sole owner of the *freezes* event in the last beat as no other author touched that event or participants within.

Relations. Relations are bipartite graphs that characterize the correlation between two dimensions in the story repository.

Author-Participant Relations. This graph visualizes the relations between authors and participants, as depicted in Figure 7 (b). The edge width between an author node and a participant node corresponds to the number of times an author has used the participant over all revisions of the story. In the exemplary repository Bob edited the participants *Scientist Horton* and *Scientist Victor*.

Author-Beat Relations. This graph visualizes the relation between authors and beats. The edge width between an author node and a beat node corresponds to the number of times an author has edited the beat or events and participants within the beat over all revisions of the story. The graph in Figure 7 (c) indicates that Alice and Bob were both editing the first two beats and Bob and Charlie were editing the last beat.

Author-Event Relations. This graph visualizes the relations between authors and events types. The event instances from all revisions are collected and then grouped by event type. The edge width between an author node and an event node corresponds to the number of times an author has edited the event type over all revisions of the story. The graph in Figure 7 (d) indicates that Bob was clearly editing the most different event types.

Participant-Event Relations. This graph visualizes the relations between participants and event types. The author instances and participant instances from all revisions are collected and then grouped by event type and participant type, respectively. The edge width between an author node and a participant node corresponds to the number of times a participant instance occurs within an event instance over all revisions of the story. Figure 7 (e) illustrates that participant *Scientist Horton* and *Ghost Hunn* were characters that appeared in the most different event types and are likely the lead characters in the story.

Meta-Relations. Meta-relations characterize relations between entities in the same dimension through their interaction with another dimension. For example, we can characterize the relation among authors based on how they interacted with participants. It visualizes the relations between all authors based on how much they used the same participants throughout all revisions. The edge width corresponds to the number of interactions. Similarly, we can quantify the meta-relation between authors based on which events they used, or how much they edited the same objects within the beats throughout all revisions.

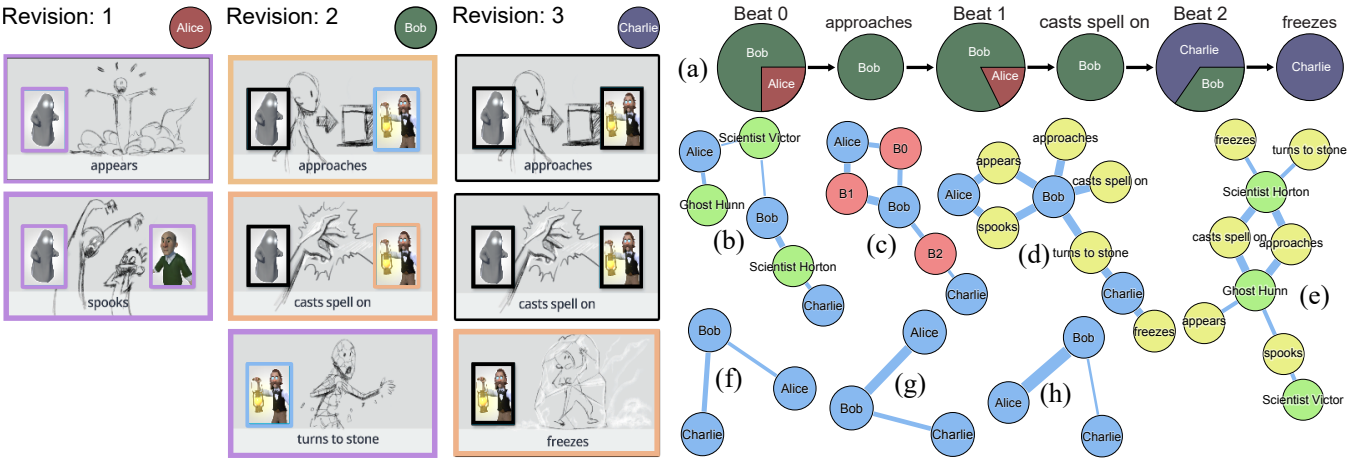


Figure 7: Left: simple story repository with three revisions and three authors, visualized using events and participants. Purple highlighting indicates inserted objects while orange highlighting indicates changed objects since the last revision. Right: different graph types generated from the repository. Beat-centric graph (a), author-participant (b), author-beat (c), author-event (d), and participant-event (e) relation-centric graphs. Graphs (f), (g), and (h) depict meta-relation-centric graphs corresponding to (b), (c), (d), respectively.

Figure 7 depicts three meta-relation-centric graphs: (f) author-participant-author, (g) author-beat-author, and (h) author-event-author, which correspond to the graphs (b), (c), and (d), respectively. Meta-relations are especially valuable for large repositories with a great number of entities per dimension, for instance, many different authors or many different event types, where direct relation graphs become extremely complex and illegible. Example meta-relation graphs for large repositories with seven authors are depicted in Figure 9. Graph (a) indicates that author Aleus is dominating and collaborating with all other authors, while the remaining authors have low connectivity among themselves. Two subgroups of authors are visible in graph (b). Authors Pallas and Orion as well as Hermes and Memnon seem to be highly engaged on the same entities of the story. Krotos is an independent author who is not editing anything that other authors have created. Finally, graph (c) conveys that all authors are strongly connected and almost all edit the same story entities.

5.1 User Study Repository Visualizations

Figure 8 and Figure 10 depict selected graphs from the repositories created as part of the user study, during which six groups (A to F) with three authors each were authoring stories using the SVC system over the course of a day. The user study is described in detail in Section 6. To anonymize the authors, their real names were replaced with a name of a Greek deity in all visualizations. Figure 8 (top) contains the story-centric graphs for each group. The author color keys are omitted for simplicity. Story lengths as well as the author's organization regarding who is assigned to which parts of the story is clearly visible. For instance, in group D, the story is divided into three parts and each part is assigned to one author. Author red made some edits in the previous part to conform it with his or her part. Analogously, the green author made edits in the first two parts. Figure 8 (a), (b), and (c) contain author-event, author-beat, and author-participant relation-centric graphs for groups A and D. A force atlas [6] layout was applied to the graphs, which arranges the nodes based on a constant gravity force as well as on the force induced by the edge widths. Additionally, the nodes are resized based on the node degrees. We found that these visual attributes increase readability greatly, as was shown in the graph visualization user study discussed in Section 6.

In group A *Scientist Horton* and *Scientist Victor* were clearly the most often used characters as is depicted in graph (a) of Figure 8. Also, their story is mostly happening in the *Room Dining* and *Room Entrance*. Graph (b) for group D confirms our observations that the authors did non share edits on the same beats. Nereus was editing the most number of beats. In contrast, the authors in group A share most of their beats. Graph (c) for group A illustrates that the authors were using many of the same beat types, which confirms again that the authors were collaborating more closely than group D.

Finally, Figure 10 depicts a participant-event relation-centric graph for group D. Various details about the story can be extracted from the graph. For instance, *Scientist Victor* and *Scientist Horton* are clearly the main protagonist of the story as they appear in the most different event types. Interestingly, *Potion Green* is investigated but never drunk. Unlike both humans, both ghosts are never frozen and never turn to stone. *Ghost Hunn* is the only ghost that both appears and disappears.

Observations. It is particularly useful to observe how the different groups worked together using the visualizations described above. We observe that group A authors were particularly collaborative, where they often worked together to iterate and finalize on the same segments of the story, and contribute to the participants story progression. This is evident in their meta-relationships where we see strong relationships develop between all three authors. The structural properties of the authored stories also emerge from these visualizations where groups A and B author complex stories with many ongoing narrative events. The other groups chose to opt for simpler, linear narratives. The authors in group D collaborate in interesting ways, as evident from their meta-relations. In terms of story beats, the authors prefer to work independently, sketching out different portions of the

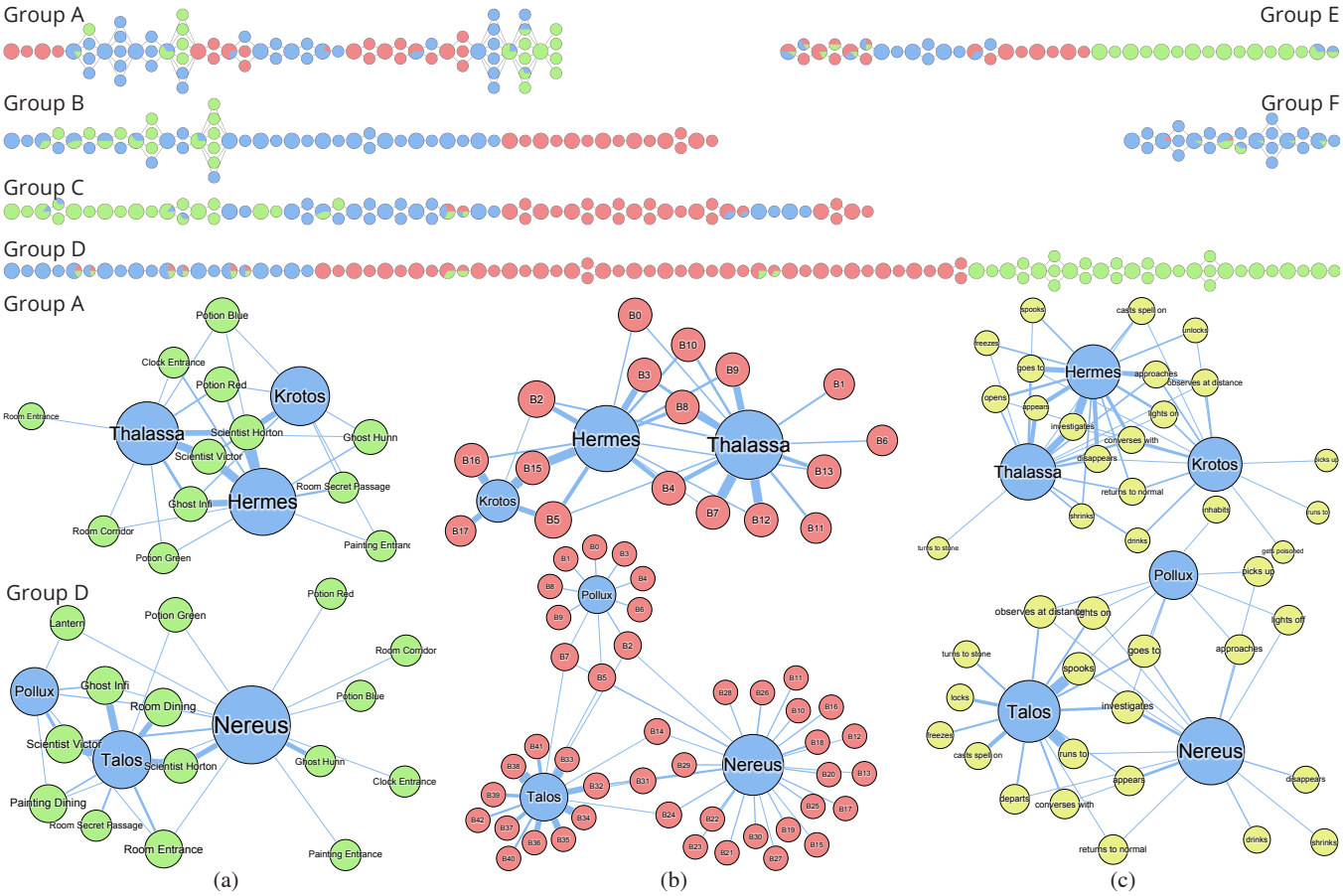


Figure 8: Selected graphs from the story repositories created during the user study. Top contains all story-centric graphs in scale. Bottom (a), (b), and (c) show author-event, author-beat, and author-participant relation-centric graphs for groups A and D.

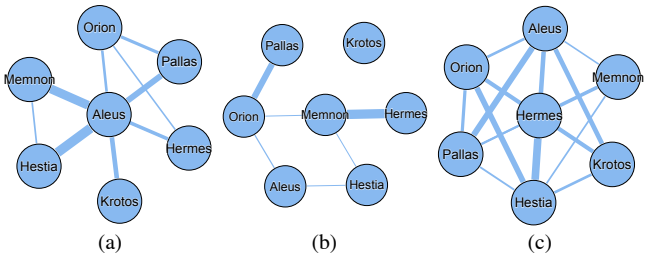


Figure 9: Meta-relation graphs characterize relations between entities in the same dimension, here authors, through their interaction with another dimension.

stories by themselves without much crosstalk. Nereus is central in determining the progression of the various participants.

6 EVALUATION

We conducted two user studies to evaluate both the usability of the system and the value of the graphical visualizations for collaboration. The first study included eighteen participants composed of 14 Computer Scientists and 4 Digital Artists. Participants were randomly grouped into 6 teams of 3 people. Each team was tasked with the goal to collaboratively co-author a story with our system. The experiment duration was 24 hours, and subjects were asked to distribute their contributions over the experiment duration. All teams were provided with the core SVC system, including functionality to author stories as visual storyboards. The user study concluded with a primary questionnaire to evaluate the entire system. A second follow-up study was conducted to evaluate graph visualizations.

6.1 System Evaluation

The primary questionnaire aimed to evaluate the usability of the system in addition to the usefulness of specific version control and visualization functionalities to support collaboration. To evaluate the usability of the system, we included the 10 question from the System Usability Scale (SUS) [3]. This questionnaire was selected because it is easy to administer and can provide reliable results for small sample sizes. Our system received a score of 74.6, which demonstrates above average usability. The SUS questions as well as the detailed results can be found in the supplementary material.

The remaining items in the primary questionnaire were selected to observe qualitative aspects about the collaborative process as

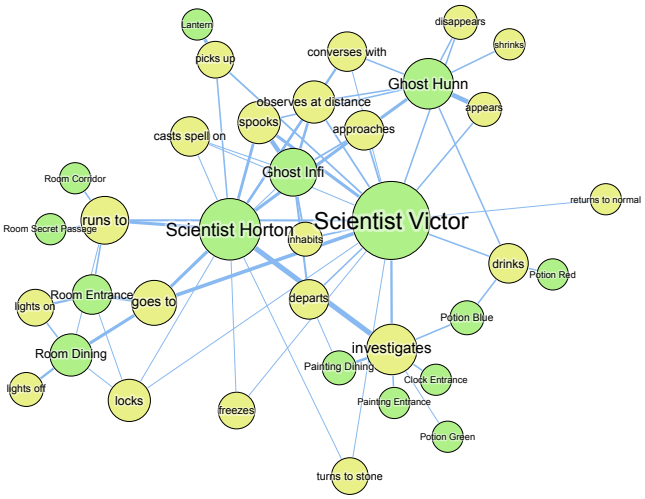


Figure 10: Participant-event relation-centric graph for group D.

well as specific components of SVC. Regarding the collaborative process, participants tended to agree that the system facilitates collaborative authoring, awareness of how co-authors were contributing and how the story evolved over time. For the basic version control functionality, subjects strongly agreed that they understood the basic functionality and tended to agree that the basic functionality operated as expected and was helpful in the collaborative process. Participants also tended to agree that they understood how to use the visual difference and update preview views, and that these functionalities operated as expected. However, the visual difference was neutrally observed to be helpful in the process of collaborative authoring. Participants tended to agree that they understood how to use the Conflict Resolution Tool and that the tool helped them resolve conflicts with co-authors. Finally, participants tended to agree that the provided version control functionality was sufficient. We generally observed that the system was useful as well as usable.

6.2 Graph Visualization Evaluation

Another aim of the user study is to evaluate the influence of graph visualizations on the collaborative process, specifically in terms of improving awareness of author participation and story content. We designed a follow-up study to specifically explore quantitative and qualitative aspects of graph visualizations. The quantitative questions provide the context for participants to analyze the graphs. After analyzing two graphs of a given type, the participant is more informed to assess whether the graph type improves awareness of author participation or story content.

We selected the 6 different types of graph visualizations: Story-centric (G1), author-participant relations (G2), author-event relations (G3), author-beat relations (G4), participant-event relations (G5), and author-beat meta-relations (G6). All graphs, except for the meta-relation-centric graphs, were generated from the completed stories authored during the first study. For each graph type, we present two graphs from two different co-author groups and ask five questions. The first question requires comparing the two graphs of given type. Questions 2 and 3 can be answered by analyzing graphs 1 and 2, respectively. Question 4 explores relevance of the graph type for supporting collaboration ("This type of graph improves awareness of author collaboration in the story."). Question 5 explores the relevance of the graph type for supporting content understanding ("This type of graph improves awareness of story content"). Please refer to the supplemental material for the complete questionnaire.

We collected responses from 15 of the original user study participants. Although some of these questions were particularly challenging, responses to quantitative questions were correct 90% of the time. The primary aim of the quantitative questions was to motivate the participant to think critically about each graph. Figure 11 summarizes the responses to qualitative questions per graph type. Subjects agreed that the story-centric authors-beats graphs (G1, see example in Figure 8-top) improved awareness of author collaboration, and disagreed that it improves awareness of story content. Interestingly, the other two graphs showing author-beats in either relation-centric (G4) or meta-relation-centric (G6) had similar responses. We interpret this to mean that visualizations involving authors and beats provide a high-level representation to improve understanding of author participation. Respondents either tended to agree or agreed that all graphs visualizing author information improved awareness of author participation. The relation-centric participant-event graph lacked author information resulting in the opposite effect; subjects agreed it improved awareness of story content.

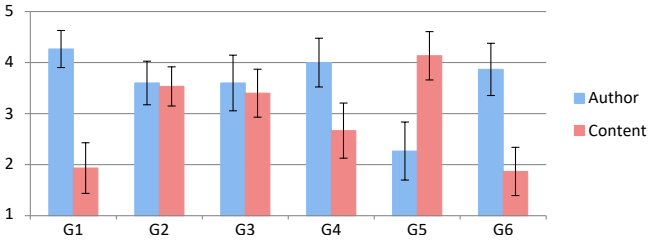


Figure 11: Graph Visualization Qualitative Questions. Responses to questions about how the respective graph improves awareness of author collaboration and story content.

6.3 Applications

We demonstrate the possibilities of our media-agnostic abstract story representation in SVC by creating stories of various media types. Figure 12 depicts three examples, including screenshots from the Haunted Castle animated story, which was automatically synthesized from our visual storyboard editor. Additionally, we created stories using (b) short movie clips as events to produce a movie and (c) photographs as narrative events to produce a comic representation. The final movie and comic rendering were manually created based on story definitions produced by our authoring system. The movie clips were extracted from the open source film Tears of Steel[8].

7 CONCLUSIONS AND FUTURE WORK

In this paper, we have presented our work to enhance collaborative story authoring by providing an SVC and graphical visualization

[8] (CC) Blender Foundation - mango.blender.org

Figure 12: Different story media types as employed by our Story Version Control system: (a) procedural animation system, (b) video clips, and (c) comic panels. Images in (b) and (c) (CC) Blender Foundation - mango.blender.org

framework. To support SVC, we propose a graph-based story representation as well as a tree edit distance operation that provides the foundation for SVC operations. We present a suite of visualizations that encode the relationships between story elements, characters, and authors, as the story evolves. These visualizations capture meaningful information that cannot be observed from a single story snapshot, or from the raw data contained in a repository, and provides a means for authors to interpret each others creative intent. Our user study validates the efficacy of our tools and our results show a variety of stories authored by untrained users, who have used our system for the first time.

Limitations. As a prototype implementation, SVC does not include production features, such as access security, distribution, or varied workflows as found in existing VCS. Further limitations in our current system motivate a number of future research directions. Currently, our system does not support branching narratives. Several users in our user study requested this feature. While we have not implemented such functionality, it could be incorporated via additional engineering effort. Users also requested commenting (annotations to communicate with other authors) and spoken character dialog. Incorporating these features could be achieved by including an additional layer on top of our system that relies on more traditional text-based version control. Directly coupling textual representations with our graph-based representation would provide an interesting area of future work.

Future Work. Additional limitations hint at more far-reaching future work. Timing in our system is not represented explicitly but defined implicitly by the given story beats. A more robust and flexible timing model could provide an additional level of control to story authors. In our work, conflict resolution operates only at the syntactic level when tree operations fail. Exploring semantic, rather than syntactic, conflicts is an exciting research direction. Such semantic detection could flag anachronisms such as a character appearing in a story after he or she has passed away. Detecting, communicating, and offering resolution suggestions for such semantic issues is a challenging future direction. Even more exciting future work could focus on extending semantic understanding to provide deeper assistance to story authors, such as suggesting portions of the story that could be most interesting to develop further.

REFERENCES

[1] Kerstin Altmanninger, Martina Seidl, and Manuel Wimmer. 2009. A survey on model versioning approaches. *International Journal of Web Information Systems* 5, 3 (2009), 271–304. https://doi.org/10.1108/17440080910983556

[2] E. E. Beck. 1993. A Survey of Experiences of Collaborative Writing. In *Computer Supported Collaborative Writing*, Mike Sharples (Ed.). Springer London, London, 87–112. https://doi.org/10.1007/978-1-4471-2007-0_6

[3] J. Brooke. 1996. SUS: A quick and dirty usability scale. In *Usability evaluation in industry*, P. W. Jordan, B. Weerdmeester, A. Thomas, and I. L. Mclelland (Eds.). Taylor and Francis, London.

[4] Jonathan D. Denning and Fabio Pellacini. 2013. MeshGit: Diffing and Merging Meshes for Polygonal Modeling. *ACM Trans. Graph.* 32, 4, Article 35 (July 2013), 10 pages. https://doi.org/10.1145/2461912.2461942

[5] J. Hart. 2013. *The Art of the Storyboard: A filmmaker's introduction*. Taylor & Francis. https://books.google.ch/books?id=3WfmPt0gbagC

[6] Mathieu Jacomy, Tommaso Venturini, Sebastien Heymann, and Mathieu Bastian. 2014. ForceAtlas2, a continuous graph layout algorithm for handy network visualization designed for the Gephi software. *PloS one* 9, 6 (2014), e98679.

[7] A. Jhala and R. M. Young. 2010. Cinematic Visual Discourse: Representation, Generation, and Evaluation. *IEEE Trans. on Computational Intelligence and AI in Games* 2, 2 (June 2010), 69–81. https://doi.org/10.1109/TCIAIG.2010.2046486

[8] Mubbasir Kapadia, Seth Frey, Alexander Shoulson, Robert W. Sumner, and Markus Gross. 2016. CANVAS: Computer-assisted Narrative Animation Synthesis. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA '16)*. Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, 199–209. http://dl.acm.org/citation.cfm?id=2982818.2982846

[9] Riccardo Mazza. 2009. *Introduction to Information Visualization* (1 ed.). Springer Publishing Company, Incorporated.

[10] S. McCloud. 1994. *Understanding Comics*. HarperCollins. https://books.google.ch/books?id=tUwqbo48lp4C

[11] Mateusz Pawlik and Nikolaus Augsten. 2011. RTED: A Robust Algorithm for the Tree Edit Distance. *Proc. VLDB Endow.* 5, 4 (Dec. 2011), 334–345. https://doi.org/10.14778/2095686.2095692

[12] Alberto Piacenza, Fabrizio Guerrini, Nicola Adami, Riccardo Leonardi, Julie Porteous, Jonathan Teutenberg, and Marc Cavazza. 2011. Generating Story Variants with Constrained Video Recombination. In *Proceedings of the 19th ACM International Conference on Multimedia (MM '11)*. ACM, New York, NY, USA, 223–232. https://doi.org/10.1145/2072298.2072329

[13] Mark O. Riedl and Vadim Bulitko. 2013. Interactive Narrative: An Intelligent Systems Approach. *AI Magazine* 34, 1 (2013), 67–77.

[14] Edward Yu-Te Shen, Henry Lieberman, and Glorianna Davenport. 2009. What's Next?: Emergent Storytelling from Video Collection. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '09)*. ACM, New York, NY, USA, 809–818. https://doi.org/10.1145/1518701.1518825

[15] K. Zhang and D. Shasha. 1989. Simple Fast Algorithms for the Editing Distance Between Trees and Related Problems. *SIAM J. Comput.* 18, 6 (Dec. 1989), 1245–1262. https://doi.org/10.1137/0218082

[16] Zhenpeng Zhao, Sriram Karthik Badam, Senthil Chandrasegaran, Deok Gun Park, Niklas L.E. Elmqvist, Lorraine Kisselburgh, and Karthik Ramani. 2014. skWiki: A Multimedia Sketching System for Collaborative Creativity. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '14)*. ACM, New York, NY, USA, 1235–1244. https://doi.org/10.1145/2556288.2557394