Define "Authoring Tool": A Survey of Interactive Narrative Authoring Tools

Daniel Green, Charlie Hargood, and Fred Charles

Bournemouth University, UK
{dgreen, chargood, fcharles}@bournemouth.ac.uk

Abstract. The creation of interactive narrative is often supported by authoring tools - either to unlock the creation process for non-technical authors or simply make authorship faster and easier. From early Hypertext focused creation tools this has now evolved to a broad diaspora of tools from specialist applications for particular types of story to generalist information system creation tools than can be used for narrative purposes, and from academic research software to commercial tools aimed at authors and game companies. In this paper we seek to answer "What is a tool, anyway, in the context of authoring for interactive storytelling?". Our results demonstrate four populations of tools - two large tools differentiated by support for key features, a smaller cluster of tools from a single publisher, and a fourth cluster of outliers.

Keywords: Authoring Tools · Interactive Narrative · Software Survey

1 State of the Art

What is a tool, anyway, in the context of authoring for interactive storytelling? Authoring tools are often at the center of the conversation for this community, as evidenced by the existence of the AIS workshop itself [2]. However from the early beginnings of Hypertext creation systems such as HyperCard [5] to more modern systems specialized for very specific forms of storytelling [3] this community is now faced with a diaspora of authoring tools of great variety. In this work we stop to consider the current state of the art in authoring tools in order to better understand the spread of applications available and the variety of both these tools functionality and the interface paradigms employed. The purpose of this is to *identify common features of authoring tools, the existence of clusters or "genres" of tool, and the spread of different approaches for supporting interactive storytelling.*

We sampled a total of 29 tools. 14 were sourced from academically published research. 4 are developed and sold as commercial products. The remaining 11 come from other non-commercial, non-academic sources, such as open-source or otherwise free projects. We took the original tool list from our previous work in this space as detailed in our earlier work [1]. This was a result of a thorough review of work from the hypertext, and interactive narrative research communities, as well as tools used by the broader creative communities in this space.

2 Methodology

Given our list of tools we wanted to appraise the feature sets of these tools to see if they could be defined by their functionality and interface design. It should be noted that while every effort was made to retrieve and use the tools themselves in order to make our appraisal, in some cases the tools were not available. Furthermore as no money was spent for this survey free demo versions were used of commercial tools where available, but this was not always the case. In the case of a tool not being available to us due to no free operational version being available features were inferred from research papers or published materials available. In the event a feature for a specific tool could not be ascertained the tool was treated for the purposes of this study as not supporting that feature.

Our feature list represents a combination of pertinent system and UI features for the tools. This list of features represents a composite of different descriptors we were able to use to distinguish between the systems themselves on a functionality or interface level. Note, this means it does not include very base level features which were identical for all systems (and consequently do not distinguish between them) nor does it include conceptual features of the underlying story models that these systems edit as for the purposes of this study we were concerned more with the tools and interfaces themselves. While there is not room in this short paper to describe all our features in detail a summary is as follows (a detailed explanation is available on request):

1. **Error Handling***: Linting, at build time, or at runtime
2. **Highlight Syntax/autocomplete***: Highlight functions or autocomplete
3. **Launcher/Dashboard**: Internal project/story management
4. **Node View**: Graph based story editing
5. **Can Duplicate Content**: Easy duplication/copy and paste
6. **Structural Shortcuts**: Shortcuts to creating particular story structures
7. **Autolayout**: Automatic tidying/structuring of content
8. **Link Parking**: Functionality for temporarily connecting content
9. **Source Editor**: Text based editing through scripting or mark up
10. **Content Browser**: Easy browsing of story content such as a film strip
11. **Searchable/Filterable**: String based content searching
12. **Relationships Method***: Visual, event based, or internal relationships
13. **Statistics**: Story stats for analytics
14. **Editing Method***: Main content editing method - either modal popups, inline fields, or sidebar inspectors
15. **Can Preview**: Internal story previewing
16. **Simple Debugging**: Variables/consoles for runtime debugging
17. **Modify During Debug**: Modifiable variables during runtime debugging
18. **Platform**: Standalone, web, mobile, or integrated into 3rd party app
19. **Documentation/Examples***: Availability of documentation/examples
20. **File Format***: JSON, XML, GBLORB, HTML, or Custom Format
21. **Able to Export**: Author can export to a desired format
22. **Exports to Runtime**: Exports for use in other applications

While most features represent a boolean as to whether the system does or does not support the feature some have varied methods of support (those marked with a star in the list) such as Editing Method (which might describe any of 3 principle interfaces for editing the text). For the purposes of subsequent analysis these were broken down into enumerated separate boolean features (e.g. Platform was broken into Standalone, Web, and Integrated booleans).

Each tool was systematically reviewed against these features through a combination of usage (where possible) and review of relevant documentation and research papers. Where usage was employed this was done on either an macOS or Windows machine as needed, with preference for macOS where both were supported. Each variable was recorded as supported or unsupported with "lacking evidence" cases recorded as unsupported.

Having collected data as to whether each tool supports or does not support these features we can perform a statistical analysis on the data set to explore any clusters or trends that might emerge from the survey. This was all done using `R` and related packages. As the data we were dealing with is purely categorical, we used Multiple Correspondence Analysis (MCA) followed by Hierarchical Clustering on Principal Components (HCPC) from the `FactoMineR` [4] package to determine relationships and clustering. We asked HCPC to identify four clusters, which was chosen based on the suggested cluster count calculated by inertia gain generated by the algorithm.

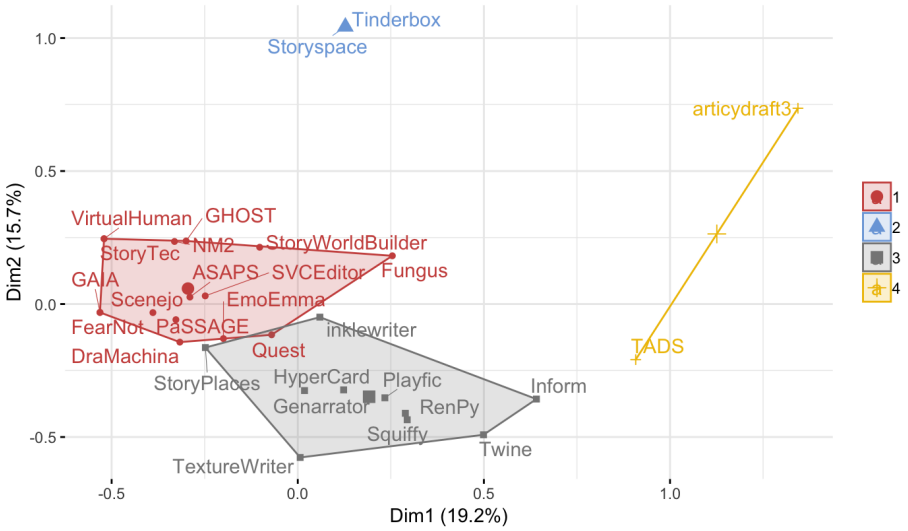

Fig. 1. Planar projection view of the 3D HCPC clustering algorithm results.

3 Results and Analysis

For 12 of the 29 systems we lacked operational systems or had limited documentation that led to incomplete evidence for some variables. We have chosen to still include these tools for completeness in our analysis and where evidence of a feature was unavailable we have treated it as unsupported rather than impute it. This affected: ASAPS, DraMachina, FearNot, GAIA, GHOST, NM2, PaSSAGE, Scenejo, Story World Builder, StoryTec, SVC Editor, Virtual Human.

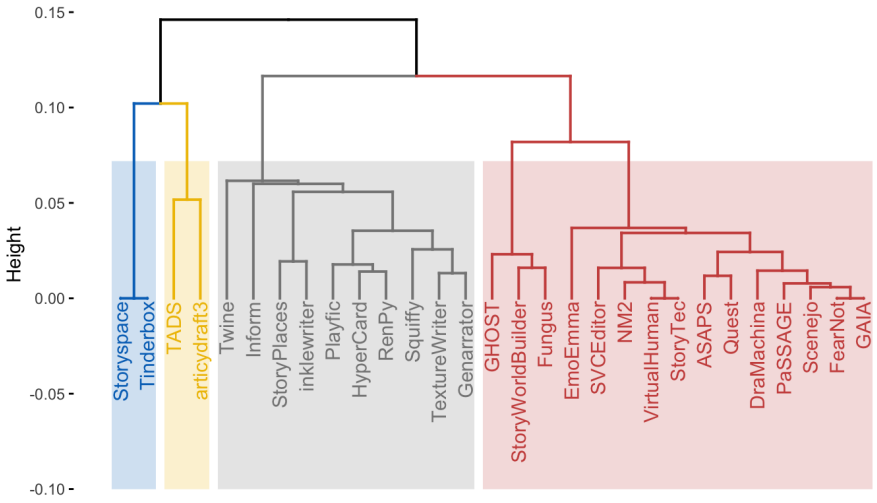

Fig. 2. Dendrogram of HCPC result clusters 1 (Red), 2 (Blue), 3 (Green), 4 (Yellow).

A planar projection of the endpoints of the 3D tree generated by HCPC can be seen in Figure 1, and a dendrogram of the HCPC clustering results can be seen in Figure 2. We can use the dendrogram tree structure and height value to determine the potential result of increasing clusters. For instance, increasing clusters to five would split the rightmost group unevenly in two. However, as the inertia gain generated by HCPC suggested four clusters as optimal, we did not perform this further partitioning.

Closer inspection of the HCPC variables helps to identify the key factors that lead to the clustering of those tools. Table 1 displays a series of tables demonstrating the key factors for each cluster along with the percentage of the overall population with this feature in the cluster and the percentage of the cluster that matches this feature. Some key common identifiers are:

– **Cluster 1**: Inspector based editing, lack of examples or documentation, XML based, not web-based, unable to export

- **Cluster 2**: Link parking, autolayout and autocomplete
- **Cluster 3**: Web applications, launcher present, Non inspector editing (mostly inline), examples and documentation
- **Cluster 4**: Runtime export, debug editing, build time error handling

Cluster	Factor	Overall%	Cluster%
1	Using XML	100%	53%
	Not being able to preview	90%	60%
	Not having documentation	85%	73%
	Editing using Inspectors	81%	87%
	Not editing using Inline	74%	93%
	Not having a launcher	71%	100%
	Not being able to export	68%	87%
	Not having a custom format	68%	87%
	Not having a web app	67%	93%
	No syntax highlighting	64%	93%
2	Being able to park links	100%	100%
	Can autolayout content	66%	100%
	Can autocomplete	66%	100%
3	Having HTML format	100%	30%
	Web application	88%	70%
	Not having a content browser	75%	60%
	Having a launcher	75%	60%
	Editing using Inline	70%	90%
	Not editing with Inspectors	69%	90%
	Not having a dynamic node view	50%	90%
4	Being able to export to runtime	100%	100%
	Having error handling at Build time	66%	100%
	Being able to make changes during debug	40%	100%

Table 1. Simplified for each cluster. Only key contributing variables are listed.

It should be noted that these are trends and not true for every member of a cluster. We can see some interesting patterns emerging however such as the two large clusters (1 & 3) while being relatively close to each other broadly speaking (as depicted in the planar diagram in Figure 1) are differentiated by XML based Inspector editors with little example, documentation, or lack of a launcher, and web based non-inspector editors with launchers and no documentation (upon further analysis). This might represent a difference in type of system, many of the systems in cluster 1 for example are academic research projects which might explain poor support in some areas where research projects have not had the resources to provide this. In contrast cluster 3 represents many of the "modern" breed of web based authoring tools that are perhaps better supported. It should also be noted however that all of the systems for which some of the data was incomplete due to it being unavailable appear in cluster 1 - while this was only for a handful of variables its possible this has distorted the data into having a

cluster of tools partly defined by their absence of evidence. If this is not the case however, it might still be said that the reasons data was not forthcoming for some of these tools is precisely because the degree to which they are supported is lacking.

The smaller clusters are arguably much easier to identify. Cluster 2 is simply the two authoring tools produced by Eastgate - they are both highly supported commercial tools with a very particular graph based style couched in Hypertext and they are very similar to each other. Its consequently not a coincidence that they would form their own cluster. Cluster 4 is the opposite - in that these are not very similar similar systems (as evidenced in Figure 1) but have been clustered through some limited similarity (such as runtime exporting) as "the outsiders" by virtue of not belonging to another cluster. We can see in Figure 1 the clear similarity between TADS and Inform (the classic text adventure tools), but by virtue of its other features the clustering algorithm has included Inform in Cluster 3 and TADS in 4.

4 Conclusions and Future Work

We present the outcome of a survey of the current state of the art of authoring tools and how these can be clustered based on features into four populations. Some initial conclusions can be drawn about these populations in that the level of support available clearly distinguishes two of the large clusters, as well as a dichotomy between inspection based non web tools, and inline editor web tools.

Our future work in this area plans to explore more closely the impact of using different tools on the authoring experience and resulting narrative, this survey serves as a starting point for identifying what can currently be considered "an authoring tool" and what might be representative tools to select in order to cover the state of the art. This data may also be of further value in future work seeking to identify more detailed patterns in this work and trends for likely future tools.

References

1. Green, D., Hargood, C., Charles, F.: Contemporary issues in interactive storytelling authoring systems. In: Proceedings of the 11th International Conference on Interactive Digital Storytelling. ICIDS 2018 (2018)
2. Hargood, C., Mitchell, A., Millard, D.E., Spierling, U.: Authoring for interactive storytelling workshop. In: International Conference on Interactive Digital Storytelling. pp. 405–408. Springer (2017)
3. Hargood, C., Weal, M., Millard, D.: The storyplaces platform: Building a web-based locative hypertext system. In: Proceedings of the 29th ACM Conference on Hypertext and Social Media. ACM (2018)
4. Lê, S., Josse, J., Husson, F., et al.: FactoMineR: an R package for multivariate analysis. Journal of statistical software **25**(1), 1–18 (2008)
5. Smith, T., Bernhardt, S.: Expectations and experiences with hypercard: a pilot study. In: Proceedings of the 6th annual international conference on Systems documentation. pp. 47–56. ACM (1988)