Syntropize*

Re-imagine the Desktop

Rahul Gupta†

Abstract

Syntropize re-imagines the desktop as a creative space from where you can plot your way to world domination. Or brainstorm the next big idea. Or make a collage of cat memes ;-). Syntropize is a free and open-source cross-platform hypermedia environment to build and share content and knowledge – a modular and hackable Integrated Development Environment for your creativity. For more, visit www.syntropize.ml.

1 Introduction

Syntropize re-imagines the desktop as a creative space for individuals as well as teams to create and share content and knowledge. Syntropize is a free and open-source, cross-platform hypermedia environment for users to capture and organize their thoughts as they engage in creative tasks or information triage. Syntropize keeps track of ideas and their inter-relationships, giving users the freedom to discover hidden patterns within their content and build new knowledge. Users are able to return to what they are working on after a day, a month or a year without any loss of context. Syntropize allows you to share content together with the associated context by simply copying or linking, making it a perfect platform for publishing and collaboration.

Syntropize is a small step towards the dream of bringing cognitive augmentation [1, 2] to everyone, right on their desktop. Syntropize variously has the capabilities of a Personal Knowledge Base, Content Management System, Information Visualization Environment, Note Taking Software, Mash-up Composition Tool, File Browser and Hypermedia Application Platform amongst others. But none of these labels are able to fully capture the essence of Syntropize; I'll leave it to the academics and experts to determine what it really is.

> In the following, we discuss the concepts underlying Syntropize. Only an essential core of this vision has been implemented in the prototype, which can be downloaded from www.syntropize.ml. Please refer to section 4: Syntropize 0.x (page 6) for more information about the pre-alpha implementation.

1.1 Express Intent, Context and Meaning

First and foremost, Syntropize lets you capture the essence of your creative endeavours.

Users capture meta-information about their creative endeavours using annotations. Annotations can range from simple "post-it" style notes to picture thumbnails, audio and video snippets, web content and/or even interactive media. Users create new annotations by selecting the appropriate annotation

*Whitepaper Draft v1.0; revised July 30, 2016.

†E-mail: rahul@syntropize.ml

template and specifying the data and/or state. Annotation templates are full-fledged web apps that communicate with Syntropize through a public API. Unlike traditional client-side web apps, annotation templates need not be bound to a particular web service/server.

When the user creates a new annotation or copies over an existing annotation on to their desktop, Syntropize stores the name of the annotation template together with the data/state specified by the user alongside the user's files. When the location is accessed, Syntropize reads back the contents in order to load the specified annotation templates for that location; it then passes the data/state to each instance of these templates, thereby regenerating the annotations.

Annotations templates can be either installed locally on a user's computer or invoked from a centralized repository/app-store. An annotation template is implemented as a set of web-components, with at least one web-component each for specifying the input and for rendering the output. Thus, it is relatively straight forward for users to extend existing templates or create their own. Users can even share these templates with other Syntropize users through a centralized repository/app-store.

1.2 Specify and Identify Relationships

Syntropize not only allows you to capture your thoughts but lets you organize them as well.

Structure and inter-relationships between annotations is expressed using visualizations. To create a visualization, the user selects a visualization container and specifies the associations between annotations using the information visualization technique implemented by that container. Information visualization techniques could include a cork-board, mind-map, network diagram, map, graph, interactive images, technical diagrams etc. to name just a few. The visual representation thus created, allows users to analyze data, organize thoughts, build stories and/or convey meaning. Visualizations encourage users to examine their content from different perspectives to reveal undiscovered patterns and identify tacit knowledge embedded in the content.

Visualization containers are hypermedia applications that transclude annotations. Much like annotation templates, visualization containers also communicate with Syntropize through a public API. They might be thought of as 'app of apps', that is, they are apps within which annotations are hosted. Data associated with visual representations is stored alongside user files as well.

Again, visualization containers can be either installed locally or invoked from a centralized repository/ app-store. A visualization container is also implemented as a set of web-components, though input and output functionalities might be typically combined. Extending them, though, is a bit more involved than annotation templates due to a more complex public API. But the benefit of this complexity is that it also provides an opportunity to create complex hypermedia applications as we shall see in subsection 1.7: Bonus: Hypermedia Applications (page 3).

1.3 Publish and Collaborate

Syntropize makes sharing your creative endeavours as simple as it can possibly be.

Users can share what they have built by simply making a copy. When the copy is opened with Syntropize, it regenerates the annotations and visualization with the result that another user will see the same content as the author.

Alternatively, users can put their work on a server (such as by copying over content on to a server or making changes to their content directly on a server from within or outside Syntropize) and share the link. This scenario is not only great for hosting content but ideal for collaboration. As users modify their content online, those changes are instantaneously reflected to all other Syntropize users at the same location. Thus, users are able to share their creations in real-time without the hassle of building a website or separately uploading files and explanatory notes to an online service. Certainly, no more e-mails with modified attachments being sent back and forth.

1.4 Build Content

Syntropize encourages you to build content organically.

When a user finds an interesting annotation, they can bring it over to their own desktop by simply copying it or linking to it. Under the hood, Syntropize copies the annotation data/state together with all the included files and recreates the annotation for the user. Other users can, in turn, use annotations created by this user as part of their work. Thus, we have a virtuous cycle of content generation where users are able build on each-others labour to the benefit of the entire Syntropize community.

1.5 Compile Projects

Syntropize lets you build and manage complex projects from your desktop.

Whether putting together a report, creating a video or writing code, a common task during most creative activities is the need to combine or compile files in some way. Syntropize allows users to select files underlying annotations and invoke scripts on these selected files to build content.

Users are therefore able to focus on individual concepts and their relationships with each other; they can selectively bring their content together depending on task that they wish to accomplish or audience they wish to reach out to. This further enhances the virtuous content cycle of content generation as users can easily combine their own creations with content that they pull-in from other locations.

1.6 Search Friendly

Syntropize makes your content more discoverable.

With Syntropize, search engines will be able to discover the intent, context and meaning behind a user's creative endeavours – information that is not captured using semantic data. By matching a query with this meta-information, search engine will be able to deliver more relevant results to its users. The author benefits as their content is more readily discovered by those who might find it the most useful.

1.7 Bonus: Hypermedia Applications

Syntropize opens new possibilities for users and developers to create hypermedia applications. Novice users will be able to create simple apps with minimal programming inputs.

Since visualizations are hypermedia applications with a public API implemented inside a web-component, it is possible for them to do more than just express structure and relationships between annotations. In fact, they can be used to perform just about any task that another web app can and some that web apps still cannot.

By wrapping a `<webview>` inside a visualization container that renders `HTML` files, Syntropize effectively becomes a web browser. In other words, the web browser is just one of many apps that can be built using Syntropize. Going beyond, Syntropize makes it easy for developers to create local and network applications using web technologies.

Novice users, on the other hand, will be able to use pre-developed containers to create simple hypermedia applications without the need to code. Such containers would tie in the functionality of annotations that the user would include and configure in order to create custom applications, each offering a unique experience of its own. Some preliminary ideas for such apps might include dashboards, forms, wizards, storyboards etc. Users can always modify the code in the container to achieve more complex functionality.

2 Features

By abstracting away the complexity of managing their thoughts and ideas, Syntropize is designed to allow users to focus on their creative tasks. At the same time, Syntropize is also designed to minimize the complexity of development, so that users and developers alike are able to adapt and extend Syntropize according to their own needs. Features that help Syntropize accomplish these goals are as follows:

2.1 Storage Agnostic

Syntropize transcends the divide between local and online content to provides you with a consistent and seamless experience.

Syntropize can interface with the different storage providers, whether it is the file-system on the user's computer or a database on a web server, through storage plug-ins. A storage plug-in converts the interface presented by the storage provider to the standardized format used within Syntropize. Based on the scheme specified in the URL, Syntropize will invoke the required storage plug-in and connect to the store.

2.2 Real-Time Synchronization

All users at a particular location see the same content, even as you make changes to it, allowing for near-instant publishing and collaboration.

Syntropize uses asynchronous events to communicate with storage providers. All changes made to the content are updated (near-)instantaneously by Syntropize, with the result that at any time, all users at the same location see identical content.

2.3 Transparent Storage

Syntropize stores user data transparently over a storage provider, so that you always have a simple mental picture of how your content is organized.

The annotation and visualization data is stored in ordinary directories (or equivalent container available with the storage provider) alongside the user's files. Subdirectories are used to hold all the annotations within that visualization, together with the content associated with those annotations. Subdirectories, in turn, can be associated with their own visualizations. In this way, Syntropize leverages the hierarchical structure of the file-system to create a transparent storage model.

2.4 No Lock-in

With Syntropize, you are free to work on your files with any software of your liking.

Since data is stored transparently, Syntropize does not convert user files in any manner before sending them to storage. Unlike some applications/platforms, the users are never locked-in. For instance, instead of using Syntropize, users can view locally stored files using a file manager.

2.5 Application Hooks

Syntropize supports the integration of external applications so that you can perform tasks with minimal disruption to your workflow .

Syntropize provides developers with hooks to integrate external applications into the user-interface. By incorporating these applications into Syntropize, users have the freedom to perform tasks with minimal

distraction. User are not required to navigate away from their content or open a new application in a separate window. Application hooks let developers create toolbar items, context menus, notifications, trigger side panels and/or dialog boxes, run scripts and/or open external applications. This feature could be used to incorporate scripts for task automation, content building, revision control etc. to cite a few examples.

2.6 Module Reuse

Syntropize allows you to reuse its modules in your own applications.

Syntropize employs interchangeable modules to provide various functionalities. These modules are based on open standards and a public API to encourage their reuse. They are not tied in any manner to Syntropize and are as such installed independently. Developers can load the same modules from the user's installation into their own applications, reaping the benefits of reduced package size, fewer network calls and improved performance. Such reuse also serves as an additional incentive for developers to create modules for Syntropize.

An important motivation for reuse is to give developers the opportunity to let their users copy annotations and visualizations from Syntropize to their applications and vice-versa.

2.7 Hosted Modules

Syntropize also supports loading of modules from a CDN/App-Store.

If the user does not want to install a module locally, it can be loaded directly from the CDN/app-store. The package manager also connects to the app-store to keep local modules up to date.

3 Design

At the heart of Syntropize is a high-level application framework that employs a three-layer architecture to provide a clear separation of concerns. Each layer in the framework acts as an independent shell with a public API. Framework layers dynamically invoke interchangeable modules to provide specific functionality. Such decoupling dramatically reduces the design complexity, creating an extremely flexible and extensible system. In particular, developers are able to build and maintain each functional component in isolation.

Syntropize comes with a package manager which is completely independent from the framework. Modules with standardized interface together with independent package management is meant to encourage reuse that goes beyond Syntropize.

3.1 Back-end: Access Layer

The storage layer enables Syntropize to connect with the required storage provider.

The storage layer establishes a connection to the storage provider in order to create a standardized data and events interface. Based on the URL scheme, Syntropize invokes the storage plug-in required to connect to the storage provider. The storage plug-ins act as a convertor for the interface presented by the storage provider.

The storage layer might also redirect through a service worker to allow for local persistence of user data. This is to allow Syntropize to work with a given online location even when the user is offline.

3.2 Middleware: Translation Layer

The translation later creates the data/events model for annotations and visualizations.

Translation layer interacts with the common storage interface to provide data and events according to the model needed by the presentation layer. This additional layer of abstraction relieves annotations and visualization from the responsibility of interpreting raw data and events. It also allows different visualizations to possibly reuse the same data/event model.

3.3 Front-end: Presentation Layer

The presentation layer acts a shell for the web-applications that create the user experience, in particular, annotations and visualizations.

The navigation bar is the only permanent user-interface element. The remainder, such as the menu, panels, dialogs etc. are created based on the location and data/event model. The presentation layer also generates the user-interface elements for applications that integrate through hooks.

More importantly, the layer acts as a shell for annotations and visualizations. It loads the web-components implementing the visualization container and the annotation templates based on the model from the translation layer (and visualization request on the URL) and instantiates them with the required data/state. It also loads the input web-components whenever the user wants to create and modify annotations and visualizations.

3.4 Package Manager

Syntropize loads modules from the independent package manager.

Whenever Syntropize requires a module, it talks to the package manager. The package manager queries a local registry of installed modules that it maintains. If the application is not installed locally, it queries the CDN/app-store. If the module is available, it downloads the package and if required, installs it as well. Finally, the package manager loads the module to memory, so that it becomes available for use.

4 Syntropize 0.x

Syntropize 0.x is a proof of concept implementation that provides the users with multimedia cards to capture their thoughts. Simply create cards in lieu of folders and organize you files in them as usual.

Syntropize 0.x is a proof of concept (pre-alpha) implementation of some of the core ideas behind Syntropize. The application framework for Syntropize is implemented using nwjs (Chromium + nodejs). Syntropize 0.x is a free and open-source software released under the Mozilla Public Licence v2.0. It is a cross-platform application that runs on Windows, Linux and OS X.

4.1 Annotation: Cards

Syntropize uses cards to provide annotations. Cards are a natural choice because of their concise yet versatile nature, which has made them the most ubiquitous user-interface paradigm [3, 4, 5] used in most large scale web applications.

A small library of text and media card templates is included with Syntropize 0.x to give the user an idea of the richness of annotations possible. From a developer perspective, a card template comprises of

two web-components, one for specifying data/state and the other for presentation. The instance meta-information is stored in a standard format inside a `json` file. This makes it relatively straightforward for developers to create new card templates.

4.2 Visualization: Board

Syntropize 0.x lets users organize cards on the screen using a bulletin-board metaphor. The board provides facilities to the user to express structure and patterns using spatial and visual cues such as the position and colour of cards. Such spatial hypertext [6] has been shown to be an effective means of describing vague and fluid relationships. It is certainly possible to have other arrangements of cards such as tabs, accordions, stacks etc. These are, however, left to future implementations.

4.3 Local Storage: File System

As discussed earlier in subsection 2.3: Transparent Storage (page 4), Syntropize is unique in that it is built directly on top of the file-system. Cards are directories with a hypermedia user-interface that contain `json` files storing the associated meta-information. The desktop workflow is not affected at all; where the users sought to create a directory, they now create a card instead. In this way, information about the content is captured with minimal disruption to the user experience. Furthermore, user's content is not locked in some proprietary manner; files and directories can be opened with locally installed software (such as the file browser).

4.4 Cloud Storage: Server with Web-Sockets

Syntropize 0.x not only works with the file system, it can also connect to a server that communicates using Chokidar events over Socket.IO. Sharing is as simple as uploading the directory to the server and sharing the URL with other Syntropize users. In the future, storage plug-ins will make it possible to connect to other storage platforms provided that changes are communicated in real-time.

4.5 Build System

Syntropize allows the integration of external software directly into the user-interface. As an example of this capability, Syntropize 0.x integrates scripts to merge or compile files that have been selected through the user-interface. Such tasks might have previously required users to open a program and specify files to merge/compile in its own monolithic window. Alternatively, users would be required tediously type in the command together with all the file paths on a prompt or create a build file that does the same. Integration of the build functionality into the user-interface minimizes such disruptions to the user's workflow.

5 Some Ideas for the Future

One of the ways in which Syntropize reduces complexity is by making it easier for users to perform tasks. To this end, some additional features that could be incorporated into Syntropize in the future are proposed below:

5.1 Command Line Interface

A Command Line Interface integrated within Syntropize will make it simple for you to perform complex tasks and allow for a simpler graphical interface.

Since Syntropize is interfacing with the file-system, support for a command line interface should not come as a surprise. Syntropize has a modular design which abstracts away the graphical menu from user commands, into which the CLI can hook in as well. A command line interface creates the opportunity to relieve the graphical user-interface of complex tasks.

More interesting are the possibilities of closer integration of the CLI with the GUI. In the proposed system, for example, Syntropize could execute tasks on files selected within the GUI, relieving users of the need to manually type filenames. Navigation could trigger a simultaneous change of directory at the prompt. Such a setup would, for example, make redundant the need to list directory contents.

Command line interface is also important because it serves as an intermediate step to natural language input and voice commands.

5.2 Support for Alternative Input Methods

Syntropize needs to be ready for the world where it is possible for you to interact with your devices using multiple input methods.

A benefit of using annotations and visualization is that they are larger and more reminiscent of physical objects, which already makes them more suitable to touch and gestures than icons or window controls.

Syntropize presently supports simple touch navigation and commands. In the future this could be extended, for example, to voice and gesture recognition through Web Speech API and Gestures.IO respectively.

5.3 Collaborative Editing

To provide you with a true community experience, Syntropize will need to go beyond just collaborative content generation and support shared editing.

Users with write access to a common location can already make changes that everyone else can see. However, no conflict resolution or revision control mechanism is provided because most storage providers, in particular, the file-system does not support these features completely. Furthermore, users see changes only after they have been made and not while they are being made.

To remedy this, Syntropize would need to add support for data stores that implement file locking and/or operational transforms. A collaborative user-interface could be implemented with the help of libraries such as TogetherJS.

6 Conclusion

Syntropize re-imagines the desktop as an **Integrated Development Environment** for your **creativity**. The desktop is envisaged not just as an interface to manage files or to serve as a dashboard, but as a hypermedia environment that let you capture intent, context and meaning behind your creative endeavours. Syntropize makes it simple for you to share content and work collaboratively, and easier for search engines to discover what you have made. Designed to be infinitely customizable, Syntropize empowers you to develop new modules to meet the needs and desires of your users. Syntropize transcends the traditional dichotomy between the browser and the desktop, creating a coherent and unified experience irrespective of the source of your content. By relieving you of the cognitive burden of keeping tabs on your content, Syntropize gives you the freedom to enjoy what you do best – *create and innovate!*

7 Appeal

Syntropize and all its components are free and open-source. I have tried to build a working prototype to the best of my meagre abilities, but it is far short of vision laid out here. To see the light of the day, Syntropize needs the love and nurture of the user and developer community. Only your enthusiasm and efforts can sustain Syntropize and create a vibrant ecosystem around it. It is my sincerest hope that each and every one of you will embrace Syntropize and make it your own. I would like to take this opportunity to **Thank You** for taking an interest in Syntropize.

If you have an idea, an opinion or a criticism, please do not hesitate to share them. Even if I am unable to follow it up every time or there is an occasional lack of agreement, it is greatly appreciated that you care and have taken the time and effort to share your thoughts. These inputs make Syntropize better for you and everyone else in the community.

References

[1] Joseph CR Licklider. Man-computer symbiosis. *IRE transactions on human factors in electronics*, (1):4–11, March 1960.

[2] Douglas C. Engelbart. Augmenting Human Intellect: A Conceptual Framework. Technical Report AFOSR-3233, Stanford Research Institute, Menlo Park, California 94025, USA, October 1962.

[3] Benedict Evans. Twitter, canvases and cards, June 2013. http://ben-evans.com/benedictevans/2013/6/18/canvases.

[4] Chris Tse. Stacking the Card Deck, June 2014. https://vimeo.com/100662919.

[5] Nick Haley. Guardian beta · The container model and blended content – a new approach to how we present content on the Guardian, June 2014. http://next.theguardian.com/blog/container-model-blended-content/.

[6] Frank M. Shipman, III and Catherine C. Marshall. Spatial Hypertext: An Alternative to Navigational and Semantic Links. *ACM Comput. Surv.*, 31(4es), December 1999.