Método de trabajo | Introducción

Si una idea domina por sobre las demás en el extenso epígrafe anterior esa es la necesidad metodológica que el videojuego, como proyecto, demanda a su creador. Con lo visto sobre los cíclicos errores en los que la industria ha caído y cotejados con el desarrollo independiente solitario y sus propios quebrantos habituales, en este apartado se propondrá SCROLOG1.0. Esta será, y sólo será, una forma de verbalizar el método propio de trabajo y en ningún momento deberá entenderse como una propuesta metodológica formal o presuntuosamente normativa. A fin de cuentas, condenar la frecuente aparición de este comportamiento en la industria videolúdica para después caer en su mismo error sólo serviría para ahondar en él, además de ser tremendamente pretencioso debido a la falta total de experiencia.

Hay que remarcar que SCROLOG1.0 es una vía para sintetizar los conocimientos obtenidos del estudio anterior, sobre todo volcado a las peculiaridades del desarrollo solitario. El objetivo no es otro que crear una propuesta propia e interna, de cara a tener un punto de partida metodológico tanto en este proyecto como en potenciales. De ahí haber apellidado SCROLOG con la simbología que insufla un número de versión: la propia experiencia, con cada proyecto propio y/o aprehendida de futuros proyectos profesionales, corregirá los errores ingenuos que esta primera versión con seguridad adolece y, poco a poco, irá puliendo SCROLOG como metodología propia hasta, quizá, poder proponerla formalmente como metodología viable y útil para el desarrollador independiente solitario, especialmente el más inexperto.

[¿Mención de alguna forma a Carlos Morcillo por “animar” a llevar a cabo esta formalización y por tanto justificarla?]

Por completar.

Método de trabajo | SCROLOG1.0

Etimológicamente, SCROLOG nace de tres epónimos. El primero de ellos, SCRUM, abandera la más que evidente victoria de las metodologías ágiles en el desarrollo de videojuegos hodierno (y en el desarrollo en general, podría decirse). Aunque, como se especificará, esto no significa que SCRUM sea la única vía de SCROLOG, sino que el desarrollador podrá usar métodos de gestión de procesos alternativos con los que se encuentre más cómodo, como Kanban o incluso la Técnica Pomodoro. El segundo de ellos, SOLO, enfatiza la naturaleza de SCROLOG como metodología focalizada en el desarrollo solitario, de manera que se dedican más esfuerzos en ella a minimizar los factores de riesgo habituales en proyectos solitarios [ver solodev]. El tercer y último epónimo, LOG, pone de manifiesto que a fin de cuentas SCROLOG trata de mejorar y abultar la forma de documentar el trabajo propio, con especial hincapié en no equiparar documentación de posproducción con *post mortem* [ver tal referencia]. Como se ha visto, habitualmente los desarrolladores solitarios lo son de manera paralela a lo realmente profesional, con el aumento de riesgos a poder asumir y obteniendo un gran valor de documentar convenientemente el avance de un proyecto, fallido o exitoso, de cara al próximo (a diferencia de lo que se apuntaba en [referencia a lo de postmortem como innecesario por ser el último], pues los *post mortem* acaban siendo un triste epitafio de empresas independientes de pequeño tamaño, por no tener la oportunidad financiera de corregir sus errores en siguientes proyectos).

Método de trabajo | Herramientas y métodos usados

Por completar.

Overleaf

Overleaf es una plataforma web dedicada a la escritura y publicación de contenido académico. Está enfocada a lo colaborativo y hace uso de *\LaTeX* para la maquetación de los documentos. Se trata de una herramienta madura y confiable que facilita varias tareas rutinarias de la escritura de documentos cualesquiera y, en especial, formales.

La decisión de haber usado una solución basada en la computación en la nube, en lugar de un entorno nativo de \LaTeX en la propia computadora, se produjo tras estudiar ciertas ventajas e inconvenientes:

Productividad: evidentemente, los tiempos de compilación en herramientas como Overleaf se agrandan en magnitudes considerables. Sin embargo, se consideró que el flujo de trabajo en documentaciones de este tipo no requiere constantes compilaciones; además, la posibilidad de compilar automáticamente en todo momento ayuda a tener una versión actualizada pocos segundos antes que puede revisarse en cualquier instante, incluso visualizarse en tiempo real como podría ofrecer un entorno alojado en la propia computadora. En momentos de maquetación y modificaciones de formato, si las compilaciones continuas se hacen tediosas, se puede recurrir sin problema a una versión mínima en un proyecto alternativo de Overleaf. Este proyecto construirá su salida en tiempos muy menores al conjunto. E incluso, si sigue demorándose más de la cuenta, toda esa tarea de formateado puede migrarse a un entorno en el computador hasta dar con la salida deseada y después importarse de manera sencillísima a Overleaf. De hecho, ofrece una posibilidad muy interesante a este respecto, permitiendo vincular archivos de otros proyectos —o archivos de cualquier otro servicio en la nube— que se mantengan actualizados. Los distintos apartados podrían disgregarse cada uno en un proyecto que importase el formato, centralizado en el proyecto global, y este a su vez importar los archivos de contenido de cada uno de los subarchivos.

Supervisión: la tarea de supervisar la documentación del proyecto queda en algo tan sencillo como acceder al archivo compartido. Evita al director tener que revisarlo desde un equipo con una instalación de \LaTeX o bien tener que estar constantemente recibiendo muestras del documento final. El enlace es ubicuo, unívoco y asegura que la versión a la que se acceda estará actualizada, incluso a tiempo real si se están realizando modificaciones en el proyecto.

Integración con otras herramientas en la nube: escribir un buen documento de estas características no es trivial. Muchas fuentes de trabajo distintas conviven en su formación y deben converger de la forma más productiva posible. Una de estas fuentes es la gestión de referencias bibliográficas. Gracias al uso de herramientas en la nube y de haber automatizado las exportaciones y actualizaciones \todosee{ver Zotero y tal}se mantuvo la naturaleza ubicua de la que Overleaf dota al proyecto.

Control de versiones intrínseco: Overleaf interioriza tanto la tarea de realizar copias de seguridad, como la manutención del proyecto actualizado en un repositorio remoto, como la distinción de versiones entre las que poder comparar archivos y restaurar en caso de que sea necesario.

Zotero

Los estándares de cita habituales hacen aguas a la hora de amparar proyectos artísticos, y más aún videolúdicos. La mayoría de ellos sigue incluyendo al videojuego como un producto de *software* convencional, negándole su condición y con ello varias de las necesidades de citación que demanda. Zotero es un gestor libre de referencias bibliográficas, frente a otros de licencia privativa o propietaria como EndNote o Mendeley. Su condición libre le otorga unas ventajas que casan a la perfección con lo anterior:

Flexibilidad: aunque los gestores de referencias suelen dar muchísimas opciones al usuario, Zotero va un paso más allá al dejar no sólo esa plétora de opciones sino muy distintas formas de adaptarse al método de trabajo de quien lo usa.

Atención a la comunidad: los creadores de Zotero están atentos a lo que sus usuarios demandan en el foro oficial. En el caso que ocupa, ayudaron a quien firma con varias cuestiones referentes a la cita videolúdica o artística en general —fílmica—.

Capacidad de modificación: el usuario que se enfrente por primera vez a Zotero con intención de escribir sobre videojuegos verá que algunas de las demandas más básicas siguen estando ausentes. No obstante, la posibilidad de añadir fácilmente complementos palia de forma sublime esta situación, como evidencia el siguiente apartado.

BBL

En conjunción, la dupla Zotero y Better Bib(La)TeX actúa como un propulsor del usuario hacia el control absoluto de su colección bibliográfica. Tanto es así, que cuesta pensar en el primero sin la adhesión del segundo, al menos a la hora de exportar en ciertos formatos. Y es que Better Bib(La)TeX amplía ciertas características de Zotero enfocadas a la exportación, sobre todo, en formatos legibles para los paquetes biblatex y biblatex.

Han sido estas las que terminaron por decantar la balanza hacia el uso de Zotero:

Exportación automática: una bendición de cara a trabajar con cambios en la colección bibliográfica mientras se añaden o modifican en el propio documento de \LaTeX{}. En resumen consiste en enlazar la colección a ciertos archivos exportados en el formato deseado, de manera que se actualicen estos archivos con cualquier cambio realizado a la colección. Si se usa una distribución de \LaTeX{} instalada en el equipo, bastará con ubicar esos archivos en las carpetas adecuadas del proyecto.

Podría pensarse que al trabajar en la nube (ver Overleaf) se pierde esta funcionalidad tan interesante, pero nada más lejos de la realidad. La magia ocurre cuando se elige para ubicar estos archivos autoexportados alguna carpeta del sistema con sincronización en la nube. En el caso que ocupa, la referente a los servicios de Google Backup & Sync. Mediante tal, se consigue que un archivo de bibliografías esté actualizado en la nube. Si sumado a eso se usa la funcionalidad ya mencionada de Overleaf para vincular archivos en la nube al proyecto, el resultado es que la bibliografía de Overleaf se mantiene actualizada en la nube respecto a la bibliografía de Zotero.

Personalización de etiquetas clave: las etiquetas clave son el identificador con el que paquetes como biblatex permiten la referencia a bibliografía. Por defecto, los gestores de este tipo generan etiquetas horribles y nada mnemotécnicas, lo cual enturbia mucho el trabajo o al menos la escritura fluida si se quieren añadir al vuelo las referencias. Mediante Better Bib(La)TeX el usuario puede elegir el formato de estas etiquetas e incluso su tratamiento en caso de conflicto. Por ejemplo, puede escogerse que el formato sea el de autor seguido del año de publicación, que resulta en etiquetas intuitivas por su similitud con los propios formatos de cita habituales. Se pueden incluso personalizar dentro de los formatos elegidos según se requiera, o desvincular cualquier etiqueta a la automatización.

Código a ejecutar tras la exportación: esta es sin duda la característica que más potencia aporta a Zotero. Mediante Javascript*\todochange{formato}*, el usuario puede añadir código que se ejecutará sobre el conjunto a exportar. No debe confundirse esto con código que peinará el documento exportado, lo cual reduciría ciertamente su atractivo; el código trabaja sobre la representación interna que Zotero estampa al momento de exportar.