Del producto descartable al patrimonio cultural: por qué el software debe preservarse como memoria técnica y social

Mauricio González Prendas

Durante mucho tiempo el software ha sido evaluado casi exclusivamente por su rendimiento práctico. Un programa parece valioso mientras funciona, produce resultados y puede reemplazarse por una versión más nueva cuando cambian las necesidades técnicas o comerciales. Esa lógica ha favorecido una cultura de actualización permanente que suele presentar la obsolescencia como algo natural, cuando en realidad también produce pérdida histórica. Si una parte importante de la vida contemporánea transcurre dentro de programas, plataformas, motores de juego, sistemas de gestión y entornos virtuales, entonces la desaparición de esos programas no es solo un asunto técnico, sino también cultural.

Esta postura encuentra respaldo en la investigación sobre preservación de software, ya que Matthews et al. (2010) demostraron que preservar software no equivale a guardar archivos aislados, porque el comportamiento de un programa depende de componentes, contextos y propiedades que deben seguir siendo inteligibles en el tiempo. En una línea semejante, Haux et al. (2021) plantean que el patrimonio digital obliga a pensar la memoria de la era informática desde marcos culturales más amplios, lo cual refuerza la idea de que el software no debe entenderse únicamente como una herramienta funcional, sino también como una forma de registro cultural. Desde esa perspectiva, mi tesis es que el software debe preservarse como memoria técnica y social porque en él se inscriben formas de trabajo, juego, aprendizaje, administración y creación que ninguna captura puramente documental logra sustituir por completo.

El software como memoria cultural

Considerar el software como patrimonio no significa afirmar que todo programa posea el mismo valor histórico, sino reconocer que muchos sistemas condensan experiencias colectivas irrepetibles. Esto puede verse con claridad en los videojuegos y otros medios interactivos. Guttenbrunner et al. (2010) sostienen que los videojuegos forman parte del patrimonio cultural digital y muestran que su preservación exige mucho más que almacenar un soporte físico o una copia binaria,

ya que el juego depende de hardware, reglas, interfaz, documentación y condiciones materiales de uso. Lo que se intenta conservar no es una cosa inmóvil, sino una experiencia configurada por tecnologías concretas.

El mismo problema aparece en los entornos virtuales complejos, como muestran McDonough y Olendorf (2011) al estudiar Second Life, pues archivar un mundo multiusuario supone enfrentar no solo dificultades técnicas, sino también restricciones contractuales, dependencia de servidores y cambios continuos en la interacción social. Ese caso ayuda a entender por qué el software no puede reducirse a una secuencia de instrucciones desligada de su contexto, ya que un programa importante deja huellas en hábitos, comunidades, lenguajes de trabajo y prácticas culturales, de modo que preservarlo implica resguardar también las condiciones que hicieron posible su significado.

Preservar software no es guardar archivos, sino conservar contextos

Uno de los errores más comunes en este debate consiste en imaginar que la preservación termina cuando se deposita una copia en un repositorio, aunque la literatura especializada muestra que esa idea es insuficiente. Woods y Brown (2010) explican que la emulación puede ser una estrategia eficaz para recuperar ejecutables heredados, pero advierten que su viabilidad depende de disponer de suficiente información sobre los entornos de origen, mientras que Hsu y Brown (2011) agregan que el análisis de dependencias es indispensable cuando se pretende recrear materiales digitales antiguos, porque sin bibliotecas, componentes auxiliares y relaciones de compatibilidad, un programa puede sobrevivir como archivo y desaparecer como experiencia ejecutable.

A esta dimensión técnica se suma una dimensión documental, ya que Jones et al. (2017) defienden la identificación persistente y la citación del software como requisitos fundamentales para describir, rastrear y reutilizar programas a lo largo del tiempo. Su planteamiento es decisivo porque muestra que la preservación también depende de que el software pueda ser reconocido como objeto intelectual y referenciado de manera estable, algo que se relaciona con lo señalado por Molina Salinas (2023), quien plantea que la preservación del patrimonio digital requiere metadatos consistentes, normalización e interoperabilidad. En otras palabras, conservar software exige mantener la relación entre código, contexto, descripción,

entorno y acceso, ya que la pérdida de cualquiera de esas capas debilita la memoria que el sistema podía transmitir.

Obsolescencia, propiedad intelectual y bien común

La obsolescencia del software no es un simple efecto colateral del progreso, sino que también funciona como una forma de amnesia institucional y cultural, porque cuando un formato se abandona, una plataforma cierra o un sistema deja de ejecutarse en el hardware actual, la sociedad pierde acceso a parte de su experiencia registrada. Esa pérdida se vuelve más grave cuando la preservación choca con regímenes rígidos de propiedad intelectual, como muestran McDonough y Olendorf (2011) al señalar que, en mundos virtuales comerciales, los límites contractuales pueden impedir acciones necesarias para el archivo. Por eso, la pregunta no es si la propiedad privada existe, sino hasta qué punto puede bloquear por completo la continuidad de la memoria digital.

A mi juicio, el software debe pensarse como un bien con dimensión patrimonial incluso cuando tenga propietarios legítimos, una idea que no elimina los derechos de autor, pero sí cuestiona que esos derechos agoten la discusión pública. González-Barahona et al. (2023), al estudiar el conjunto de licencias preservadas en Software Heritage, muestran la importancia documental que tiene registrar el marco jurídico de los programas dentro de los esfuerzos de conservación. Ese dato permite defender una posición equilibrada: la creatividad y la innovación necesitan incentivos, pero una cultura completamente cerrada termina volviendo inaccesibles las herramientas con las que una sociedad produjo conocimiento, ocio, administración y memoria.

Qué debería preservarse

No todo software puede preservarse con el mismo nivel de detalle, pero eso no invalida la tarea, sino que obliga a establecer criterios históricos y culturales para priorizar aquellos programas que transformaron prácticas sociales, educativas, científicas o artísticas, así como los sistemas que representan comunidades, instituciones o momentos técnicos significativos. En algunos casos bastará con conservar código fuente, documentación y capturas de uso, mientras que en otros será necesario mantener ejecutables, dependencias, manuales, registros de

interacción y entornos emulados, por lo que la decisión correcta dependerá del tipo de software y del valor que se pretenda transmitir a futuro.

Por eso la preservación no debe orientarse por la acumulación indiscriminada, sino por una lógica de memoria pública, ya que preservar software también significa decidir qué queremos seguir entendiendo de nuestra propia historia digital. Allí reside la diferencia entre ver el código como residuo reemplazable o como patrimonio cultural, porque cuando se asume esta segunda perspectiva, cambia también nuestra relación con la tecnología: dejamos de pensarla solo como consumo inmediato y empezamos a verla como parte del archivo vivo de la vida contemporánea.

El software merece ser preservado porque articula memoria técnica, prácticas sociales y experiencia cultural, y la investigación revisada muestra de manera consistente que la conservación de programas no puede reducirse al almacenamiento de archivos, ya que su supervivencia depende de contextos de ejecución, dependencias, metadatos, identificación persistente y marcos de acceso. Además, también evidencia que la obsolescencia no es neutral, porque borra huellas de trabajo, creación y sociabilidad que una sociedad digitalizada no debería perder sin reflexión crítica.

Por esta razón, preservar software implica cambiar la forma en que entendemos la tecnología, pues el código no debería verse únicamente como un producto descartable, sino también como una obra cultural capaz de conservar parte de la memoria social de una época. Aunque el software pueda estar protegido por derechos de propiedad intelectual, esa condición no elimina su dimensión pública ni su valor patrimonial, de modo que las tensiones entre creatividad, innovación y propiedad intelectual no justifican el abandono de estos materiales, sino que exigen diseñar mecanismos que permitan conservarlos con criterios históricos, técnicos y públicos de largo plazo.

Referencias

González-Barahona, J. M., Montes-Leon, S., Robles, G., & Zacchiroli, S. (2023). The software heritage license dataset (2022 edition). *Empirical Software Engineering, 28*(6), 147. https://doi.org/10.1007/s10664-023-10377-w

Guttenbrunner, M., Becker, C., & Rauber, A. (2010). Keeping the game alive: Evaluating strategies for the preservation of console video games. *International Journal of Digital Curation, 5*(1), 64–90. https://doi.org/10.2218/ijdc.v5i1.144

Haux, D. H., Maget Dominicé, A., & Raspotnig, J. A. (2021). A cultural memory of the digital age? *International Journal for the Semiotics of Law, 34*(3), 769–782. https://doi.org/10.1007/s11196-020-09778-7

Hsu, A., & Brown, G. (2011). Dependency analysis of legacy digital materials to support emulation based preservation. *International Journal of Digital Curation, 6*(1), 99–110. https://doi.org/10.2218/ijdc.v6i1.175

Jones, C. M., Matthews, B., Gent, I., Griffin, T., & Tedds, J. (2017). Persistent identification and citation of software. *International Journal of Digital Curation, 11*(2), 104–114. https://doi.org/10.2218/ijdc.v11i2.422

Matthews, B., Shaon, A., Bicarregui, J., & Jones, C. (2010). A framework for software preservation. *International Journal of Digital Curation, 5*(1), 91–105. https://doi.org/10.2218/ijdc.v5i1.145

McDonough, J., & Olendorf, R. (2011). Saving Second Life: Issues in archiving a complex, multi-user virtual world. *International Journal of Digital Curation, 6*(2), 89–108. https://doi.org/10.2218/ijdc.v6i2.192

Molina Salinas, C. (2023). Desafíos de la preservación digital del patrimonio cultural en México: el caso de Mexicana. *Cuadernos.Info*, (55), 211–232. https://doi.org/10.7764/cdi.55.48751

Woods, K., & Brown, G. (2010). Assisted emulation for legacy executables. *International Journal of Digital Curation, 5*(1), 160–171. https://doi.org/10.2218/ijdc.v5i1.150