

الترحيل الدائم: نقل الأنظمة ممارسة مستمرة في المكتبات الرقمية

إليزابيث ماكولي



المخلص:

تنتقل هذه الورقة الفكرية من تجربة برنامج المكتبة الرقمية في مكتبة جامعة كاليفورنيا في لوس أنجلوس، التي تخوض مشروع ترحيل واسعًا متعدد الجوانب، لتدعو إلى النظر إلى ترحيل الأنظمة بوصفه نشاطًا مستمرًا مدمجًا في العمل اليومي، لا حدثًا منفردًا أو مشروعًا كبيرًا. تعرض المؤلفة دورة حياة الأنظمة البرمجية في المكتبات من الاختيار إلى التنفيذ فالاستخدام فعدم الرضا ثم الترحيل والإيقاف، وتسرد تاريخ الأنظمة التي استخدمتها المكتبة منذ 2004: من نظام مخصص لكل مجموعة، إلى مستودع مطور محليًا، ف (Islandora)، ثم (Samvera) مع نهج الخدمات المصغرة المبني على (IIIF). وتقدم مبادئ للترحيل تشمل التوثيق والشفافية والعناية بالبيانات وعدّ التشغيل البيئي فعليًا، وتؤكد إعادة فحص الغرض من النظام، وأهمية الجرد، والاستعداد للترحيل التالي. وتخلص إلى أن الأنظمة يديرها أشخاص، وأن تبني عقلية «الترحيل الدائم» يتوافق مع واقع دورات حياة الأنظمة.

الكلمات المفتاحية: ترحيل الأنظمة، دورة حياة الأنظمة، المكتبات الرقمية، الخدمات المصغرة، جرد البيانات

مقدمة:

تقوم معظم أعمالنا في المكتبات على أنظمة برمجية. ومع ذلك، ومهما بلغ رسوخ نظام ما في أنشطتنا اليومية، فإن جميع الحلول البرمجية مؤقتة. ولكل نظام برمجي داخل المؤسسة دورة حياة: (1) الاختيار، و(2) التنفيذ، و(3) الاستخدام، و(4) عدم الرضا، و(5) الترحيل والإيقاف (decommission). فهناك مرحلة صعود النظام: الاختيار والتنفيذ. ثم تأتي مرحلة الاستقرار التي نستخدم فيها النظام استخدامًا روتينيًا ويكون فيها جزءًا لا يتجزأ من عملنا. لكن سرعان ما يدخل النظام مرحلة الأفول، ويصبح قرب انتهاء عمره مصدر قلق. وقد تعجل بمرحلة الأفول قوى داخلية أو خارجية، مثل عدم رضا المستخدمين و/أو المطورين عن النظام، أو قرارات إنهاء دعم البرمجيات (end-of-life) الصادرة عن جهات خارجية. وحين يبلغ الضغط الخارجي أو الداخلي حدًا حاسمًا، تبدأ المؤسسات دورة حياة جديدة لنظام برمجي. وفي مقالهم الحديث، يقدم بابكوك وآخرون (Babcock et al. 2020) عرضًا عامًا جيدًا لدورة الحياة هذه

فيما يخص نظام البرمجيات الخاص بمجموعاتهم الرقمية. وحين يكون أحد الأنظمة في طور الأفول، تتولد عادةً دورة حياة أخرى تفضي إلى ولادة نظام جديد، ومن أجزاء عملية تبديل الأنظمة هذه المهمة المربكة والمرهقة: الترحيل. والترحيل هو النشاط الذي يجري لمرة واحدة وننقل فيه محتوانا من النظام القديم إلى النظام الجديد.

دورات حياة الأنظمة البرمجية جزء طبيعي من مؤسسات المكتبات، ومن ثم فإن عمليات الترحيل نشاط متكرر في مجموعات تقنية المكتبات. وأرى دليلًا على شيوع أنشطة الترحيل في مؤسستي، وكذلك عند الرجوع إلى المقالات المنشورة في هذه المجلة. ففي وقت كتابة هذا المقال، تعمل مكتبة جامعة كاليفورنيا في لوس أنجلوس (UCLA) على ترحيل أربعة من أنظمتنا الرئيسية: نظام إدارة محتوى الموقع العام، ونظام إدارة الموارد الإلكترونية، والنظام الآلي المتكامل للمكتبة، ومستودع المجموعات الرقمية. وعند النظر إلى مجتمع (Code4Lib) الأوسع، يتجلى تواتر مشروعات الترحيل في تنوع المقالات المنشورة حول هذا الموضوع في مجلة (Code4Lib Journal)؛ إذ إن عشر مقالات على الأقل نُشرت خلال السنوات الأربع الماضية تتناول تحديدًا أدوات الترحيل أو أساليبه أو مشروعاته. ومع ذلك، كثيرًا ما يُنظر إلى الترحيل على أنه نشاط في اتجاه واحد ولمرة واحدة، أي مجرد مشروع كبير فيه كثير من الشيفرات المكتوبة لحالة بعينها التي يُستغنى عنها بعد إنجاز المهمة. ونظرًا إلى حتمية الترحيل في عمل تقنية المكتبات، أرى أن هناك فرصة لتحديد مبادئ للترحيل تدعم المكتبات في تبني الأنظمة واستخدامها. وبينما تفيد الأمثلة والمقاربات المحددة للترحيل، يهدف هذا المقال إلى أن يكون دليلًا مرافقًا لأي عملية ترحيل. وسأقدم في هذا المقال بعض التفاصيل عن سياق ترحيل المجموعات الرقمية في جامعة كاليفورنيا في لوس أنجلوس، ثم أعرض بعض المبادئ الدائمة المقترحة للترحيل.

دراسة الحالة: الخلفية:

أنشئ برنامج المكتبة الرقمية في مكتبة جامعة كاليفورنيا في لوس أنجلوس وحدةً من وحدات المكتبة في عام 2004، وكان يتألف من أمناء مكتبات، ليعمل بالشراكة مع فريق صغير من مطوري برمجيات المكتبة الرقمية في وحدة تقنية المكتبة. ومنذ عام 2004، أنشأت وحدة برنامج المكتبة الرقمية رصيدًا متناميًا من المجموعات والمشروعات الرقمية الفريدة وحافظت عليه. بل إن بعض هذه المشروعات ما يزال قائمًا على الويب بصورته التي تعود إلى نحو عام 2004، ومن ذلك مجموعتنا «أدبيات الحملات الانتخابية» (Campaign Literature)، التي تضم مطبوعات عابرة (ephemera) متعلقة بالانتخابات في كاليفورنيا من عام 1908 إلى عام 1940. ونحن ندير في الوقت الحاضر 201 مجموعة منشورة تضم 3.25 مليون أصل رقمي و235 تيرابايت.

وخلال المدة من عام 2004 إلى الآن، استخدمنا أنظمة كثيرة مختلفة. وكما هو الحال في كثير من مبادرات المكتبات الرقمية، سعينا في جامعة كاليفورنيا في لوس أنجلوس إلى دعم المشروعات المخصصة ذات الخصائص الفريدة، وكذلك الأنظمة العامة التي تعزز زيادة الإنتاج. وقد مرت مدة ركزنا فيها على الجمع بين هذين النقيضين، باستخدام أنظمة عامة لإدارة البيانات الوصفية وملفات المحتوى وهيكلتها تحت واجهات مخصصة لعرض هذه المواد وإتاحة تجارب مستخدم فريدة. فقد احتفظت قاعدة البيانات وبرامج تطبيقات الويب الأساسية بسير عمل موحد يمكنه دعم الإنتاج الكبير والتجميع (aggregation)، ثم يمكن إضافة برمجة مخصصة لتطبيقات الويب فوقها من أجل الخصائص المتخصصة والهوية البصرية الفريدة. وقد شغلنا هذا المسعى محليًا، لكنه يتوافق أيضًا مع التطورات الأوسع في مجال المكتبات الرقمية. ففي تلك المدة، كانت مؤسسات كثيرة تطور مستودعات وتقنيات أخرى تدعم زيادة التوحيد القياسي والتشغيل البيئي: (DSpace)، و(Fedora)، و(Hydra)، و(Samvera) (حاليًا)، و(Islandora)، والإطار الدولي للتشغيل البيئي للصور (International Image Interoperability Framework, IIIF).

في البداية، اتبعنا نهج «مجموعة واحدة لكل نظام». فقد طورنا مخطط قاعدة بيانات مخصصًا خاصًا بنا، وتطبيق ويب مخصصًا خاصًا بنا، ثم كنا ننسخ المخطط والتطبيق ونخصصهما لكل مجموعة. وكانت كل مجموعة نسخة قائمة بذاتها يمكن تخصيصها بالقدر الذي تسمح به إمكانياتها. ومع تراكم المجموعات، أدركنا أن مجموع المجموعات أكبر قيمة من أي مجموعة منفردة، فأعدنا تصميم حلنا المخصص المكون من قاعدة البيانات ولغة (Java) ليصبح نظام مستودع ونشر مبنياً خصيصاً. وكان هذا النظام الجديد يعني أنه صار لدينا نظام مستودع مطور محلياً يتيح كفاءة أكبر وقابلية اكتشاف (discoverability) أفضل.

غير أننا سرعان ما بدأنا ندرك أننا معزولون بنظامنا المطور محلياً. فلم نكن قادرين على مشاركة الشيفرة مع مؤسسات أخرى، ولذلك كنا معتمدين دائماً على موظفي التطوير لدينا لتحقيق أي هدف جديد. وفي الوقت نفسه، لاحظنا أن مؤسسات أخرى تتجه نحو تقنيات مشتركة وتطور الخصائص والحلول بسرعة أكبر. وللمشاركة في هذا الأسلوب الجديد في العمل، اعتمدنا (Islandora) حلاً لمكتبتنا الرقمية، و(Drupal) نظاماً جديداً لإدارة المحتوى لموقع المكتبة الأوسع. وقد حسن هذا الحل الجديد بعض المشكلات التي كانت تحد من قدراتنا، لكننا بعد نحو أربع سنوات أدركنا مجدداً الحاجة إلى نظام جديد يلئم احتياجاتنا على نحو أفضل. وكان أول أسبابنا لاختيار نظام جديد أن التآزر المحتمل بين (Drupal) و(Islandora) لم يكن مثمراً بالقدر المتصور. وثانيها أننا أدركنا أننا لم نتواصل تواصلًا فعالاً مع مجتمع (Islandora) الأوسع من المطورين والمستخدمين.

وهكذا بدأنا في عام 2016 دورة حياة أنظمتنا من جديد. فبدأنا بعملية مراجعة واختيار رسمية لحل جديد للمكتبة الرقمية. وكلف اثنين من موظفي المكتبة الرقمية بالبحث وكتابة ورقة بيضاء (white paper)؛ وطلب منهما تحديد احتياجاتنا، وإجراء مسح للأنظمة المتاحة، وتقديم توصية بشأن كيفية المضي قدماً. واستخدمنا الورقة البيضاء، بوصفنا مؤسسة، لدعم النقاش ومراجعة الخيارات واختيار نظام جديد. واخترنا (Samvera)، وهي في الواقع حزمة من الأنظمة، وتحديدًا (Hyrax) لإدارة الأصول الرقمية، و(Blacklight) للاكتشاف (discovery) والعرض. ومنذ ذلك القرار، اعتمدنا النظام الجديد وبدأنا تنفيذه، وفي الوقت نفسه أرسينا نهج الخدمات المصغرة (microservices). وفي هذا النهج، طورنا سير عمل وحزمة من الخدمات المصغرة تتمحور تحديداً حول إعداد أصول الصور وإتاحتها باستخدام الإطار الدولي للتشغيل البيئي للصور (International Image Interoperability Framework, IIIF). وقد فصلنا تطبيقات (IIIF) لدينا إلى مجموعة من الخدمات تستخدمها تطبيقات مكتبتنا الرقمية. وتشمل خدماتنا المصغرة المتمحورة حول (IIIF) في الوقت الحاضر إعداد الصور، وإعداد بيان (IIIF-manifest)، وخدام صور (IIIF). وبعد إعداد الصور عبر هذه الخدمات المصغرة، نستورد إلى (Hyrax) ملفاً بقيم مفصلة بفواصل (CSV) يتضمن معرفات (URI) لأصل الصورة وبيان (IIIF) المقابل له. وتضيف نسخة (Hyrax) لدينا البيانات إلى فهرس (Solr) يُنسخ إلى فهرس ثاني يشغل نسخة (Blacklight) العامة لدينا. وفي هذه البنية الحالية، يمكن استبدال أي خدمة من الخدمات أو إعادة تصميمها مع بقاء المكونات الأخرى دون تغيير. وعلى غرار جامعة تورنتو، كما أورد بابكوك وآخرون (Babcock et al. 2020)، ستواصل مكتبة جامعة كاليفورنيا في لوس أنجلوس تقسيم بنيتها التحتية وأنظمتنا إلى خدمات مصغرة لزيادة مرونة التطوير والصيانة والتحسين في المستقبل. وفي إطار هذا التحول، قد نجد أنفسنا في ترحيل دائم، ولو بقدر يسير على الأقل.

يجب أن تتدفق البيانات:

والواقع أنه حتى وأنا أصارع التحديات والتفاصيل المتعلقة بترحيلنا الحالي إلى (Samvera)، فإننا نعيد ترحيل بعض المجموعات التي رُحلت بالفعل لكي نواكب الخدمات المصفورة الأحدث. وقد وجدت هذا الواقع محبطًا إلى حد ما، مع أنني أؤيد فلسفة الخدمات المصفورة. وفي خضم ردة الفعل هذه، أدركت أن هناك فرصة لتحليل أنشطة الترحيل لدينا والانتقال من عقلية غائية (telic) إلى عقلية لا غائية (atelic). فالكلمة اليونانية (telos) تعني الغاية أو الهدف، ومن ثمّ فالأنشطة الغائية تتجه إلى بلوغ نهاية محددة. وفي المقابل، فإن الأنشطة اللاغائية هي تلك التي تُمارس لذاتها ومن دون نهاية. وبينما من الواضح أن مشروع الترحيل الذي نعمل عليه في جامعة كاليفورنيا في لوس أنجلوس لمكتبنا الرقمية سيكتمل، وسيكتمل بنجاح، فإنني أود أن أحاول إعادة تصور الترحيل بوصفه ممارسة، مثل اليوغا، بدلًا من كونه حدثًا منفردًا مثل سباق أو مباراة. وقد تتضمن عقلية الترحيل الدائم التفكير في بياناتنا الوصفية ومحتوانا كالماء الذي يستطيع مديرو المستودعات توجيه تدفقه في أي وقت إلى حيث تدعو الحاجة.

مبادئ الترحيل:

تعلمت من تجربتي في عدة عمليات ترحيل أنه على الرغم من أن الترحيل من نظام إلى نظام يكون دائمًا خاصًا وغير قابل للتكرار، فإن هناك بعض المبادئ التي تقوم عليها كل عملية ترحيل وتستحق المشاركة. وأقترح المبادئ الآتية:

- التوثيق: خصص الوقت لتوثيق بياناتك ومشاركتها مع مؤسستك. وحين تتخذ قرارات بشأن كيفية تنظيم بياناتك، وثّقها كذلك. اترك سجلًا واضحًا لقراراتك، فقد تحتاج إليه لاحقًا لاستعادة خطواتك.
- الشفافية: تعامل مع عملية الترحيل كأنها شيء قد يحتاج شخص آخر إلى تولّيه في أي وقت.
- اعنِ بياناتك: الواجهات والأنظمة تتغير. أما البيانات التي تتدفق عبر هذه الأنظمة فهي القيمة الحقيقية في جميع مؤسساتنا.
- التشغيل البيئي فعل: نقل البيانات بين الأنظمة يحسّن قابليتها للتشغيل البيئي.

أولًا، توجد الأنظمة لتخدم غرضًا ما. وفي الماضي، كان ذلك الغرض والنظام الحالي متوائمين ومقتربين. غير أنه حين يقترب نظام من نهاية عمره، تكون تلك لحظة مناسبة لإعادة فحص الغرض الذي يخدمه. اطرح الأسئلة. ما الخدمة المرتبطة برسالة المؤسسة التي يتيحها النظام أو يدعمها؟ وبأي طرق تريد المؤسسة تطوير تلك الخدمة؟ وما الاختناقات التي تعوق الكفاءة أو تحط من جودة الخدمة؟ وبطرح هذه الأسئلة، يمكنك أن تضمن أنك ومؤسستك لا تخطان بين تقنية متقدمة وخدمات أو عمليات عفا عليها الزمن.

ثانيًا، جرد ما يحتويه النظام أمر أساسي. فالجرد فلسفي وعملي معًا. فمن الناحية الفلسفية، ينبغي أن يعرض الجرد التصور الذي يقوم عليه النظام: ما الأشياء التي يحتويها النظام، وكم نوعًا من الأشياء هناك، وما العلاقة بين الجرد الداخلي (وهو قاعدة بيانات في الغالب) والشيء الذي يشير إليه (ملف، أو رابط، أو كيان مادي). ويجب أن يُظهر لك جردك التصور الذي يقوم عليه النظام، ثم يجب أن يقدم لك حجم محتوياته. ويمكن أن يفيد الجرد في جميع مراحل دورة حياة النظام، وأن يساعدك على تقييم فاعلية نظامك باستمرار.

ثالثًا، لن تكون عملية الترحيل هذه الأخيرة. فكل ما يتعلق بالترحيل يتركز على إنجازه: بلوغ الحالة المنشودة التي يصبح فيها كل شيء في النظام الجديد ويعمل على نحو أفضل. ثم يمكننا العودة إلى عملنا المعتاد وسيسير كل شيء بسلاسة أكبر. وهذا التصور الشائع للترحيل صحيح جزئيًا فحسب. فالنظام الجديد سيحسّن بعض عملك المعتاد، لكنك ما

تزال في دورة حياة الأنظمة. والأهم من ذلك أن بياناتك وبياناتك الوصفية في حركة دائمة، تخدم رسالة مؤسستك. فأنت دائماً في طور تخيل طرق جديدة لتنسيق الروابط بين موادك والوصول والبيانات والبيانات الوصفية والمستخدمين والموظفين. ولذلك، إن كان ترحيل آخر قد يحدث في المستقبل القريب، فإنني أشجعك على الاستعداد له الآن. وأرى في ذلك خدمة تقدمها لنفسك في المستقبل. هل هناك جانب من جوانب عملية الترحيل هذه أو من النظام الجديد يمكنك ترتيبه لجعل عمليات الترحيل المستقبلية أيسر؟ ومن المجالات التي ينبغي النظر فيها: التوثيق، وسلامة البيانات، وعمليات الجرد، والاتصالات.

الخاتمة:

هذا المقال ورقة فكرية حول ممارسة معتادة في تقنية المكتبات: ترحيل الأنظمة. وأحسنا على أن نرى الترحيل نشاطاً يمثل مرحلة من دورة حياة، وأنا منخرطون دائماً في تلك الدورة. فعمليات الترحيل تعطيل لسير العمل وتوزيع المهام القائمين، ويكرّس لها جهد هائل، لكن هذه العمليات كثيراً ما تمتلئ بنصوص برمجية يستغنى عنها بعد انتهاء المهمة، وبخرائط مطابقة (mappings) للبيانات أو البيانات الوصفية لا تفيد إلا في عملية الترحيل الحالية. والتحدي أمامنا هو أن نستخلص من كل عملية ترحيل دروساً قد تساعدنا حين تكون التفاصيل كلها مختلفة تماماً عند حلول عملية الترحيل التالية. وأعتقد أننا بهذه المبادئ نستطيع أن نتبنى روح الترحيل الدائم التي تتوافق على نحو أوثق مع واقع دورات حياة الأنظمة في المكتبات.

تذييل:

ملاحظة أخيرة بشأن الأنظمة البرمجية: إنها تُدار بأيدي أشخاص. فأفضل حل برمجي لقائمة متطلبات مؤسستك ليس بالضرورة النظام المناسب لمؤسستك. فالنظام البرمجي واستراتيجية الترحيل يجب أن يتوافقا مع دوافع مؤسستك وقدراتها، ومع قيادة مؤسستك، ومع موظفيك الحاليين. وينبغي للمديرين وصناع القرار أن يدفعوا أعضاء الفريق إلى تحمل تحديات جديدة ومواصلة تعلم تقنيات جديدة. ومع ذلك، ينبغي أن نضمن أن يتوافق التحدي مع قيم القادة ودوافعهم. وفضلاً عن ذلك، إن تغير الموظفون الرئيسيون المرتبطون بالنظام، فخذ لحظة لتقييم ما إذا كان مسارك الحالي ما يزال يحظى بالتوافق نفسه. وحتى لو كنت قد أنهيت للتو ترحيلاً إلى نظام جديد، فلا بأس في أن تكون في ترحيل دائم. بل إنك، باتباع هذه المبادئ، مستعد له.

المراجع:

- Babcock K, Lee S, Rajakumar J, Wagner A. Where Do We Go From Here: A Review of Technology Solutions for Providing Access to Digital Collections. The Code4Lib Journal. 2020 [accessed 2020 Nov 14]; (47). <https://journal.code4lib.org/articles/15000>
- Banerjee K, Forero D. 2017. DIY DOI: Leveraging the DOI Infrastructure to Simplify Digital Preservation and Repository Management. The Code4Lib Journal.(38). [accessed 2020 Dec 29]. <https://journal.code4lib.org/articles/12870>.
- Becker D, Williamson E, Winkle O. 2020. CollectionBuilder-CONTENTdm: Developing a Static Web 'Skin' for CONTENTdm-based Digital Collections. The Code4Lib Journal.(49). [accessed 2020 Dec 29]. <https://journal.code4lib.org/articles/15326>.

- Broadwell P, Childress D. Digital Library Systems and UCLA: An Environmental Scan with Suggestions for Near and Medium-Term Technology Strategies and User Stories / Functional Requirements. Zenodo; 2017. <https://zenodo.org/record/3027359#.X-pb6-IKg1s>. doi:10.5281/zenodo.3027359.
- Gomez J, Clarke KS, Vuong A. 2020. IIF by the Numbers. The Code4Lib Journal.(48). [accessed 2020 Dec 18]. <https://journal.code4lib.org/articles/15217>.
- Hodges DW, Schlottmann K. 2019. Reporting from the Archives: Better Archival Migration Outcomes with Python and the Google Sheets API. The Code4Lib Journal. (46). [accessed 2020 Dec 12]. <https://journal.code4lib.org/articles/14871>.
- Maistrovskaya M, Newson K. 2019. Making the Move to Open Journal Systems 3: Recommendations for a (mostly) painless upgrade. The Code4Lib Journal. (43). [accessed 2020 Dec 29]. <https://journal.code4lib.org/articles/14260>.
- Mayo C, Jazairi A, Walker P, Gaudreau L. 2019. BC Digitized Collections: Towards a Microservices-based Solution to an Intractable Repository Problem. The Code4Lib Journal. (44). [accessed 2020 Dec 12]. <https://journal.code4lib.org/articles/14445>.
- Mayo D, Bowers K. 2017. The Devil's Shoehorn: A case study of EAD to ArchivesSpace migration at a large university. The Code4Lib Journal.(35). [accessed 2020 Dec 29]. <https://journal.code4lib.org/articles/12239>.
- Ramshaw V, Lecat V, Hodge T. 2018. WMS, APIs and LibGuides: Building a Better Database A-Z List. The Code4Lib Journal.(41). [accessed 2020 Dec 29]. <https://journal.code4lib.org/articles/13688>.
- Teal W. 2018. Alma Enumerator: Automating repetitive cataloging tasks with Python. The Code4Lib Journal.(42). [accessed 2020 Dec 29]. <https://journal.code4lib.org/articles/13947>.
- Van Tuyt S, Gum J, Mellinger M, Ramirez GL, Straley B, Wick R, Zhang H. 2018. Are we still working on this? A meta-retrospective of a digital repository migration in the form of a classic Greek Tragedy (in extreme violation of Aristotelian Unity of Time). The Code4Lib Journal.(41). [accessed 2020 Dec 12]. <https://journal.code4lib.org/articles/13581>.