منهجية معتمدة على البرمجة المشتقة من النماذج لتنظيف واستيراد المعطيات أثناء التشغيل

د. عمار جوخدار [1]

الملخص

يقدم هذا البحث منهجية تستعمل نماذج جديدةً معتمدة على البرمجة المشتقة من النماذج MDP. تفيد في استيراد وتنظيف البيانات أثناء التشغيل الحي بسهولة ومرونة وفاعلية، حيث تسمح للمستخدم النهائي –الخبير في منطق العمل– باستيراد وتنظيف بياناته بنفسه دون الحاجة لمهارات برمجية.

تعد عملية استيراد وتنظيف البيانات أكثر المراحل أهمية وحساسية وكلفة في النظم المؤسساتية، مثل نظم الحكومات الإلكترونية، ونظم إدارة المشافي، ونظم الحوالات المالية والمصرفية، وإدارة الكلف وغيرها. حيث يؤدي فشل هذه المرحلة إلى فشل المنظومة ككل.

تتميز منهجيتنا بقدرتها على الاكتشاف المبكر للأخطاء قبل الاستيراد الفعلي للمعطيات مع تحقق من أخطاء سلامة المعطيات والأخطاء الدلالية كما لو أن الاستيراد قد تم فعلاً. كما تسمح بالتراجع عن الاستيراد الفعلي دون العودة بقواعد البيانات إلى حالتها ما قبل عملية الاستيراد، أي التراجع عن نتائج الاستيراد فقط والمحافظة على البيانات التي دخلت بعد الاستيراد، وهذا من ضرورات العمل على برامج تعمل بشكل حي.

قمنا بتحقيق النماذج ضمن بيئة تطوير مشتقة من النماذج هي بيئة الإكسير والتي تعتمد على بنية تحتية مفتوحة المصدر J2EE وJBoss وMySQL. وجرى اختبار المنهجية على المنظومة الناتجة عن هذا التحقيق.

تم اختبار والتحقق من هذه المنهجية ضمن مشروع إدارة كلف الخدمات الصحية بالتعاون مع وزارة الصحة، حيث جرى التطبيق على مجموعة من المشافي التابعة لوزارة الصحة وهو مشفى المجتهد ومن ثم على مشفى العيون ثم قامت وزارة الصحة باستعماله بشكل مستقل تماماً في بقية المشافي.

الكلمات المفتاحية: تنظيف البيانات، استيراد البيانات، ذكاء الأعمال، بيئة عمل الإكسير، البنيان المشتق من النماذج، النظم المؤسساتية، إدارة الكلف، اللغات التصريحية، اللغات الذكية، الجداول متعددة الأشكال، الحذف الشلالي، نموذج تصميم الأرصدة.

[1] كلية الهندسة المعلوماتية، جامعة دمشق

MDP Based Approach for Data Cleansing and Importing at Runtime

[(1)]Dr. Ammar Joukhadar

Abstract

This article presents a new methodology that uses a new models based on MDP (Model Driven Programming), which allows us to build user friendly and efficient ETL/data cleansing framework that functions in live environment. It enables business experts to import and clean data themselves, as it does not require any background in programming or database management.

Data Importing and Cleansing is the most sensitive and expensive step in enterprise application development (such as e-government systems, ERPs, forex, banking, HMIS ...) and when it fails the entire project fails.

Our methodology is capable of detecting errors early, even before the actual importing of data. It can check not only integrity errors but semantic errors also (as if we have already imported our data to the live system).

Moreover, our methodology allows us to roll back the imported data, any time, without restoring the whole database (ie., users can keep any record created after the importing process and remove only data related to the imported data). This makes our model appropriate to import data to live systems.

We have implemented our approach using a MDP framework (Elixir) which is based on free/open source infrastructure including J2EE, JBoss, MySQ.

The final system is tested and validated within a real "cost control" project, which was applied for the medical services in the Ministry of Health (MoH) in Syria. The first test is achieved in the biggest hospital belonging to MoH (Mujtahed Hospital), then in Eye hospital and the team of MoH continue the work by himself in all other hospitals.

Keywords: Data Cleansing, Data importing, Business Intelligence, Elixir framework, MDP, Enterprise Applications, cost control, declarative languages, intelligent language, polymorphic table, cascade delete, balance design pattern

[(1)]Information Technology Faculty, Damascus University

1 المقدمة

تسعى معظم المؤسسات الحكومية والشركات الكبيرة والمتوسطة للإفادة العظمى من المعطيات المتراكمة مع الزمن والتي يمكن التنقيب فيها عن معلومات مفيدة جداً في تحسين الخدمات ورفع الأرباح وتوفير النفقات. وغالباً ما تكون هذه البيانات موزعة بين مجموعة من الأقسام والبرمجيات غير المتكاملة مما يدفع بهذه المؤسسات إلى العمل على جمعها في منظومة واحدة وقاعدة بيانات واحدة لاستخراج المعلومات المفيدة منها وهو ما سمي بأنظمة ذكاء الأعمال أو Business Intelligence.

يمكن أن تتم عملية الجمع من خلال نقل المعطيات إلى منظومة موحدة باستعمال مكاملة من نوع EAI (Enterprise Application Integration)، إلا أن وجود معطيات قديمة وتاريخية يحتم علينا استعمال الأسلوب الكمي أو ما يسمى (ETL (Extract Transform Load، والذي يعتمد على تصدير المعطيات من مصادرها الأساسية على شكل ملفات ثم إعادة استيرادها إلى المنظومة المركزية بهدف تجميعها واستخراج تقارير دعم القرار المناسبة [1]. الحاجة لاستيراد المعطيات القديمة ودمجها ليست قضية الأمس فقط فما تزال العديد من المعطيات تنشأ حديثا في أماكن مختلفة ويوجد فوائد متوقعة من دمجها معا ولذلك نجد أن العديد من الدراسات الحديثة تسعى اليوم لتطبيق التنظيف على المعطيات الكبيرة Big Data [2] وخاصة ما يتعلق بالقيم المفقودة [3] والشاذة [4].

اعتمدت بعض الدراسات على نماذج للمعطيات (Models) مقدمة من المستخدم [5] إلا أنها تتطلب تدخلاً بشرياً، وتستلزم تكرار التدخل البشري لبناء النموذج الدلالي ونموذج المعطيات من اجل كل مصدر جديد للمعطيات، في حين تحتفظ لغات البرمجة المشتقة من النماذج بهذه المعطيات في مرحلة البناء ولا حاجة لتكرار بنائها لاحقاً.

من جهة أخرى، لا يكفي جمع البيانات في قاعدة بيانات مركزية واحدة ومتكاملة لاستخراج تقارير صحيحة، فلا بد أن تكون المعطيات المستوردة "نظيفة" لكي تصلح لبناء تقارير دعم القرار، ومن هنا ظهر مصطلح تنظيف البيانات Data Cleansing، حيث يهدف تنظيف البيانات إلى جعلها صحيحة وكاملة ومنسجمة [6]:

- صحة البيانات (Correctness) : أي أن تكون البيانات خالية من الأخطاء النحوية والدلالية ودون تكرار. يحدث التكرار عندما نستعمل مصطلحين مترادفين للإشارة لنفس الدلالة أو الغرض، كأن يرد اسم القسم "قسم الاشعة" في جدول النفقات ويرد على أنه "قسم الصور الشعاعية" في جدول الإيرادات، وكلا الرمزين يشيران إلى نفس القسم إلا أنهما مختلفان من وجهة نظر قواعد البيانات مما سيؤثر على تقارير دعم القرار.
- اكتمال البيانات(Completeness) : البيانات الكاملة هي البيانات التي تخلو من أي نقص، كأن ينقص أسماء بعض الموظفين في جدول الرواتب أو في الذاتيات. وعادة ما يمكن التعرف على وجود النقص من خلال قواعد عمل business rules، كأن يجري التحقق من تساوي عدد الموظفين الواردين في قوائم الذاتيات مع عدد الموظفين الواردين في قوائم الرواتب، أو أن يجري التأكد من أن تبعية الموظفين للأقسام تساوي عدد الأقسام. سيؤدي نقص البيانات إلى خلل في التقارير، حيث ستبدو كتلة الرواتب أقل أو أكبر أو تبدو المؤسسة أكبر أو أصغر.
- انسجام البيانات (Consistency): البيانات المنسجمة هي البيانات التي لا تنطوي على تناقض، فقد يكون اسم القسم الوارد في منظومة النفقات مختلفاً عن اسم القسم الوارد في منظومة الإيرادات على الرغم من أنهما يحملان نفس الرمز أو الرقم، وهنا ستكون المعلومتان

متطابقتان ومتناقضتان في آن واحد، حيث أن رمز القسم وحيدٌ ولا يجوز تكراره. في هذه الحالة، لن نتمكن من استيراد المعلومتين معاُ لمعالجتهما لاحقاً كما هو الحال في تجنب التكرار.

في دراسة أجرتها شركة IDC العالمية في العام 2007، أوضحت فيها أن أكثر من نصف تكلفة نظم ذكاء الأعمال تصرف على تنظيف واستيراد البيانات[7]. ، ومرد ذلك إلى كلفة اليد العاملة البشرية الضرورية للقيام بهذا العمل والمتخصصة فيه، فمن يقوم بهذا العمل هم من المهندسين المتخصصين القادرين على التعامل مع قواعد البيانات.

تتم عملية استيراد البيانات عادة من خلال إجراءات ETL المعروفة حيث يتم تحضير ملفات إكسل أو تصدير جداول من قواعد البيانات أو البرمجيات القديمة (Extract) ومن ثم تحويلها للصيغة التي تناسب المنظومة الجديدة Transform وأخيراً يجري استيرادها للمنظومة الجديدة Load.

يمكن للمستخدم استعمال أدوات ETL جاهزة مثل IBM Visual Warehousing, Microsoft DTS, Oracle Warehouse Builder وغيرها، إلا أن هذه الأدوات تحتاج لمهارات معلوماتية من جهة، كما أنها تستهلك الكثير من الوقت والجهد. يجب على المستخدم بالإضافة لفهمه لهذه الأدوات أن يستوعب إجراءات العمل في المنظومة التي يستورد لها البيانات، كما ينبغي عليه إعادة تصميم عملية الاستيراد في حال تغيرت إجراءات العمل. تعاني هذه الطريقة من عدة إشكاليات تتمثل في:

1- إمكانية حدوث خطأ أو نقص أثناء تصدير المعطيات من مصادرها الأصلية
2- إمكانية حدوث خطأ في التحويل إلى الصيغة الجديدة أو نقص في حقول الربط لأن الربط لا يتم بنفس الآلية في النظام القديم والحديث.
3- إمكانية حدوث خطأ أثناء الاستيراد مما يؤدي إلى توقف العمليات في منتصفها ويترك قاعدة البيانات الجديدة في حالة غير سليمة تحتاج لجهود مهندسين مختصين لتصحيح قاعدة البيانات تمهيدا لإعادة استيراد المعطيات بعد تصحيحها.
4- الوقت الطويل الذي يضيع في تكرار الاستيراد بعد كل خطأ
5- الحاجة إلى مهندسين مختصين
6- الكلفة العالية بسبب الحاجة لمهندسين مختصين

يعرض Wang في [8] بيئة عمل يسعى فيها لدعم عمليات الاستيراد بطريقة ETL بهدف التخفيف من صعوبة الاستيراد والتنظيف. هذه البيئة موجهة للمطور، إلا أنها غير مناسبة للمستخدم النهائي.

بيرتوسي وآخرون [9] اقترح تقنية تنظيف البيانات معتمدة على مطابقة الارتباطات ومطابقة الوظائف والتي لا تعمل بشكل صحيح في بعض الحالات التي تجري فيها المطابقة بترتيب عشوائي. كما أن هذه الطريقة تتطلب في الكثير من الأحيان مساحة كبيرة.وعبء معالجة كبير. قام [10] باقتراح تحسين على نهج برتوسي على أساس بيئة تفسير interpretation تعمل مباشرة على قاعدة البيانات وحسن بشكل كبير من كفاءة وأداء أنظمة الاستعلام. ولكن كلا المنهجين محدود كونهما لا يأخذان منطق العلم ولا إجراءات العمل وبالتالي البعد الدلالي بالحسبان، حيث دلت دراسة أخرى [11] أنه لا يكفي الحصول على معطيات سامية توصف الأغراض لنتمكن من استيراد وتنظيف البيانات، بل لا بد من أخذ إجراءات العمل بالحسبان، حيث أن استيراد المعطيات من قاعدة إلى أخرى مباشرة لا يمر بإجراءات العمل وقواعد العمل التي تمر بها المعطيات الواصلة بالطرق الاعتيادية عبر واجهات المستخدم أو

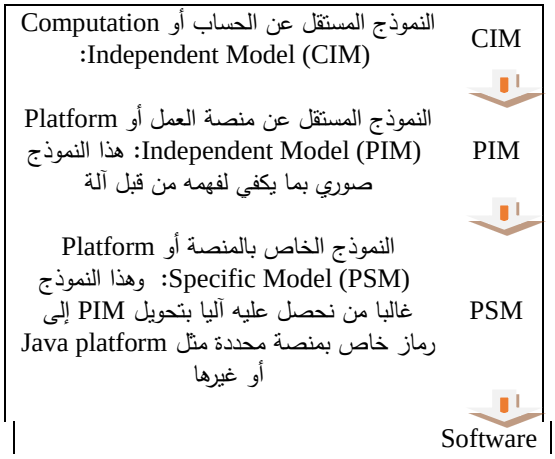
API. لذلك، فلا بد من أخذ إجراءات العمل بالحسبان في عملية الاستيراد والتنظيف.

يهدف هذه البحث إلى تقديم أسلوب جديد في استيراد وتنظيف البيانات لا يحتاج ليد عاملة متخصصة ولا إلى مهندسين وإنما يسمح للمستخدم النهائي للنظام باستيراد وتنظيف البيانات بنفسه بسهولة وسرعة عالية وقد اعتمدنا في عملنا هذا على خصائص البرمجة المشتقة من النماذج Model Driven Programming [12] [13] [14] [15] والتي تؤمن للمستخدم النهائي معطيات سامية Meta Data حول الأغراض والإجراءات تصل للمستوى الدلالي في معالجة البيانات وفرزها ومقارنتها، كما اعتمدنا على المسافة المفرداتية [16] [17] [18] بين الكلمات المرجعية المستوردة وعلى العنقدة clustering [19][20] لاكتشاف وجود كلمات مكررة بأكثر من كتابة إملائية وقمنا باقتراح الكتابة الصحيحة آلياً.

نستعرض في المقطع الثاني من هذه المقالة البرمجة المشتقة من النماذج ونعرض في المقطع الثالث المعطيات السامية التي طورناها واعتمدناها في نموذجنا المقترح، ونفصل المراحل المختلفة للمنهجية المقترحة، ثم نعرض في المقطع الرابع الحل والمنهجية المقترحة وفي المقطع الخامس النتائج التي حصلنا عليها عند اختبار النظام، ونختم بخلاصة.

2 البرمجة المشتقة من النماذج

تهدف البرمجة المشتقة من النماذج إلى تحويل البرمجة من شكلها الإجرائي إلى شكل تصريحي declarative، حيث يقتصر دور المبرمج على وضع النموذج الوظيفي للأغراض وتحديد إجراءات العمل بشكلها التسلسلي، كما هو حال BPMN أو بشكلها التصريحي كما هو الحال في دفق العمل باعتماد القواعد Rule Based Workflow.



الشكل (1) البرمجة المشتقة من النماذج

يوضح الشكل [1] البرمجة المشتقة من النماذج، والذي يتميز بالقدرة على تضمين التحليل المكتوب بطريقة الـ CIM ضمن نموذجي PIM و PSM والبرمجية النهائية بطريقة تسمح بتعقب العلاقة بين التحليل والتصميم والبرمجة والناتج النهائي. تسمى البنية التي تحمل هذه المعلومات بالمعطيات السامية أو Meta Data.

لما كان التحليل يعكس وجهة نظر المستخدم، فستنعكس –باستخدام هذه المنهجية– وجهة نظر المستخدم أيضاً في البرنامج الناتج، وستسمح له بصياغة طلباته وأوامره بطريقة تنسجم مع التحليل ومع النموذج الخاص بالمنصة. سيتيح ذلك للمستخدم النهائي أن يقوم بصياغة أوامره وأعماله دون الحاجة لخبرة خاصة في مجال إدارة قواعد البيانات أو البرمجة.

تسمح البرمجة المشتقة من النماذج وباعتماد المعطيات السامية بتوفير المعلومات الضرورية لبناء أدواتٍ لم تكن ممكنة في حال البرمجة التقليدية.

3 المعطيات السامية Meta Data

نعرض في هذا المقطع منهجيتنا الخاصة في تحقيق المعطيات السامية، والتي تسمح بربط كامل وبنيوي بين

الأغراض والإجراءات وصولاً للاستفادة الآلية من المعطيات السامية دون تدخل بشري.

نميز نوعين من المعطيات السامية: نوع لتوصيف الأغراض، وآخر لتوصيف إجراءات العمل ويمكن لأي منهما الإشارة للآخر مما يجعل نموذجنا متكامل ضمنيا.

3-1 توصيف الأغراض

توصف المعطيات السامية الخاصة بالأغراض الحقول والعلاقات بين الأغراض المختلفة. يعرض الشكل 2 مثالاً لتوصيف نموذج "غرض" وفقاً لنموذجنا، حيث يمثل كل سطر حقلاً أو علاقةً من حقول هذا الغرض، مثل الرمز ID، ونوع الحقل Type، وهل هو إلزامي أو وحيد أو قابل للتعديل، ووصفٌ كاملٌ للحقل يتضمن معلوماتٍ تفصيليةٍ، مثل القيمة الافتراضية للحقل –إن وجدت– وطريقة احتسابه عند الضرورة، وآليات الربط، وغيرها.

Trade	Code: id	Unique iid		Label : Name	Integrity :Trades with amount more than 10000 can't be created	Description:
Field name	**ID**	**Type**	**Mandatory**	**Unique**	**Editable**	**Description**
id	Id	Long	True	True	False	An Id unique to every Trade
Name	Name	String	True	False	True	The Name of the Trade which is shown to the user
amount	amount	Double	True	False	True	The amount of the trade

الشكل(2) نموذج الغرض

كما نجد في أعلى الاستمارة، معلوماتٍ خاصةٍ بنوع الغرض، مثل الرمز الوظيفي، ومجموعات الحقول الفريدة، وشروط سلامة المعطيات integrity، ووصفٌ إضافيٌ للحقل.

جميع مواصفات الحقول ليست بالضرورة أرقام أو قيم منطقية وإنما هي تعابير حاسوبية يمكن لها أن تشير إلى أغراض أخرى أو إلى خطوات معرفة ضمن إجراءات العمل workflow steps.

3-2 توصيف إجراءات العمل

تصف المعطيات السامية لإجراءات العمل (وفقا لمنهجيتنا) من قام، بأي عمل، ومتى، وكيف، وعلى أية معطيات (نوع من أمن المعطيات). هذه المعطيات السامية ما هي إلا لغة تصريحية لها مجموعة من العبارات مسبقة التعريف. كل عبارة تصف خطوة من خطوات الإجراء، مما يسهل عملية الانتقال من نموذج CIM إلى نموذج PIM. اعتمدنا في نموذجنا لتوصيف الإجراءات على مجموعة من الخطوات الهامة: 1) القائمة الرئيسية و2) المهام اليدوية و3) المهام الآلية و4) الأعمال الخلفية و5) الأعمال البسيطة simple actions و6) الأعمال التفاعلية. يوضح الشكل 3 أمثلة لبعض هذه الخطوات.

Main Entry	Accountant (**who**) can create a trade (**what**) whenever he wants (**when**) by inputting service, amount, currency and provider, then save the trade as pending (**how**). He can see only local currency (**business security**)
Manual Task	Accounting manager (**who**) can validate trades (**what**) when they are pending (**when**) by right-clicking the trade and selecting the action "validate" (**how**). He can see only trades belonging to his own agency (**business security**)
Automatic Task	The system (**who**) updates related accounts' balances (**what**) when a new trade is validated (**when**) by increasing the balance by the new amount (**how**). System has access to all trades (**business security**)
Background Job	The system (**who**) close all trades (what) at the end of the day (**when**) by creating closing transactions (**how**). System has access to all trades (**business security**)

الشكل (3) أمثلة عن خطوات نموذج الإجراء

أدخلنا، بالواقع، في لغتنا مفهوم الطرائق methods التفاعلية، أي السماح بتعريف طرائق method ضمن الأغراض والصفوف تعبر عن حالة استعمال وليس عن تابع أو إجراء بسيط، وأسميناه view action، ونعبر عنه بلغة خاصة اسمها ETDL أو Elixir Task Definition Language [21]. ومن ثم فإن هذه اللغة المعرّفة قادرةٌ على التعبير عن الأرتال الثلاثة: الزبون client، ومنطق الأعمال business logic والتخزين Storage بعبارة واحدة.

4 المنهجية المقترحة

اعتمادا على منهجيتنا في توصيف معطيات سامية متكاملة وبنيوية [21]، نعرض في هذا المقطع حلنا الذي يسمح بجعل استيراد وتنظيف البيانات في متناول المستخدم النهائي، الضليع بالبعد الوظيفي للمسألة، والذي ليس لديه أية خبرة في قواعد البيانات، أو البرمجة، أو حتى في طريقة بناء العلاقات بين الأغراض. يتطلب ذلك إيجاد حلول للتحديات التي يمر بها المستخدم دون تدخل مهندسين، وأهم هذه التحديات هي:

1- وجود خطأ من نوع السلامة integrity، مثل الخطأ في التحويل من صيغةٍ سابقةٍ إلى الصيغة الجديدة. عملية التحويل هذه قد تؤدي إلى توقف استيراد الملف في نقطةٍ قد لا يستطيع المستخدم تحديدها إلا من خلال الاطلاع على قاعدة البيانات، لمعرفة إلى أين وصل النظام في عملية الاستيراد. يتطلب حل المشكلة إما (1) إعادة الاستيراد، وفي هذه الحالة، يجب أن يكون النظام قادراً على تمييز أنه قد سبق واستورد هذه المعطيات سابقاً، كيلا يكرر استيرادها، و إما (2) المتابعة من نقطة التوقف، وهذا يتطلب من النظام القدرة على معرفة نقطة التوقف ضمن الملف المستورد، أو قيام المستخدم يدوياً بحذف السجلات التي سبق واستوردها النظام بشكلٍ صحيح.

2- نقص في السجلات: قد يؤدي النقص في السجلات المستوردة إلى أخطاء ضمن سجلات لم يجر استيرادها. فمثلاً، يمكن أن يؤدي نقص القيود المالية والمحاسبية المستوردة إلى اختلال أرصدة المالية والمحاسبية التي لم نستوردها.

3- اختلاف في حقول الربط بين الصفوف: فقد تكون العلاقة بين الصف "زبون" وبين تصنيفه مبنيةً على المعرف pk، وهو sequence داخلي في المنظومة القديمة، في حين أن الربط في المنظومة الجديدة مبنيٌ على معرف داخلي آخر هو ID، ولا يطابق في قيمته المعرف .pk. وهنا سنقترح طريقة لإعادة بناء الارتباط.

يحتاج حل هذه التحديات إما لمهندسين مختصين ضليعين في فهم منطق العمل وإدارة قواعد المعطيات للقيام بعملية الاستيراد بشكل آلي، أو لإيجاد حل مؤتمت للقضايا التي تنشأ عن الأخطاء المذكورة أعلاه. من أهم متطلبات هذا الحل المؤتمت ما يلي:

1- القدرة على الاكتشاف المبكر لأخطاء السلامة integrity والأخطاء الدلالية

2- القدرة على التراجع عن استيراد سجلٍ صحيحٍ أو خاطئ

3- القدرة على إعادة بناء الروابط

4- القدرة على محاكاة الاستيراد لاختبار ما قد ينجم عن إجراءات العمل من أخطاء دون المساس الفعلي بقاعدة البيانات الحية

سنعرض فيما يلي المنهجية المقترحة لتحقيق متطلبات الحل المؤتمت

4-1 القدرة على الاكتشاف المبكر لأخطاء السلامة والأخطاء الدلالية

يكمن التحدي في كون البرمجيات الحديثة مطورةٌ لقبول استيراد معطياتٍ من مصادر مختلفة، إذ تتكرر عملية الاستيراد عند كل إرساء جديد للبرمجيات في بيئات تمتلك مصادر معطيات مختلفة. ومن ثم فقد تواجهنا أخطاء دلالية أو أخطاء في سلامة المعطيات مختلفةٌ عن تلك المتوقعة أو التي تصل من المداخل النظامية للمنظومة. لنأخذ مثلاً حالة استيراد غرض من نوع Person، له أب وأم، وهما حقلان إلزاميان من نوع Person، فنحن لا نستطيع استيراد شخص بدون أبيه وأمه، وفي حال وجود نقصٍ لسببٍ ما، فإن الاستيراد سيتوقف بسبب رفض قاعدة البيانات قبوله دون الحقول الإلزامية، مما سيوقف عملية الاستيراد في

منتصفها، مما يستوجب تدخل خبيرٍ لتصحيح الخطأ، ومتابعة الاستيراد إن أمكن أو الإعادة.

الحل المقترح لهذا المتطلب، هو عدم التوقف عن الاستيراد بسبب وجود خطأ من نوع integrity، بل استيراد الأغراض إلى جدولٍ مؤقتٍ أسميناه الجدول متعدد الأشكال polymorphic table ليس عليه أية قيود على مستوى قاعدة البيانات، مما يتيح إتمام الاستيراد كاملاً قبل البدء بتنظيف البيانات. جميع الحقول في الجدول المؤقت غير إلزاميةٍ وغير منمطةٍ، لأن وجود نمط قد يمنع الاستيراد كأن تختلف صيغة التاريخ المستورد عن صيغة التاريخ في قاعدة البيانات على سبيل المثال.

وهنا تظهر مشكلةٌ أخرىٌ، إذ سنضطر لإنشاء جدولٍ لكل نوع غرض، إضافة إلى الجدول النهائي، بل قد يصل الأمر –حسب مصدر البيانات– إلى إنشاء جدولٍ لكل عملية استيراد، إذ قد يتضمن مصدر البيانات عدداً من الحقول أكبرَ من ذلك الموجود ضمن قاعدة البيانات الجديدة بسبب تغيير نموذج المعطيات.

في هذه الحالة يمكن استيراد جميع الملفات إلى جدولٍ وحيدٍ ومشترك، وهذا سيطرح بدوره مشكلة نقل المعطيات لاحقاً من الجدول المؤقت إلى الجدول الفعلي. هنا في هذه المرحلة يكمن دور البرمجة المشتقة من النماذج، إذ نقوم بربط معطيات ساميةٍ Meta Data ديناميكيةٍ خاصةٍ بكل سطرٍ من أسطر الجدول المؤقت مطابقةٍ للمعطيات السامية المرتبطة بالجدول العامل الموافق له، وهذا سيسمح لنا بالتعامل معها تارةً على أنها عديمة التنميط (أثناء عملية الاستيراد)، وتارةً على أنها منمطة (أثناء عملية التحقق) ولذلك اسميناه بالجدول متعدد الأشكال. ومن ثَمَّ، ستجري عملية التحقق مبكراً، قبل وصول البيانات إلى الجدول النهائي ودخولها إلى قاعدة البيانات العاملة، مما يسهل التراجعَ المبكرَ عنها في حال وجود أي خطأ كبيرٍ نسبياً يصعب التراجع عنه.

سنتمكن بهذه الطريقة واعتماداً على المعطيات السامية من التحقق آلياً من جميع أخطاء السلامة ومنها:

1- نمط الحقل: عددٌ أو تاريخٌ أو نصٌ أو قيمةٌ من مجموعة محددة من القيم.

2- صيغة الحقل: هجريٌ أو ميلاديٌ للتاريخ، بريدٌ إلكترونيٌ أو اسمُ علمٍ أو غيره للنصوص، 24 أو 12 ساعةٍ للوقت، وغيره.

3- مجال التعريف: القيمُ الصغرى والعظمى للأرقام والتواريخ، والقيمُ المحتملة للنصوص مثل الجنس أو اليوم من الأسبوع، وغيره.

4- الروابط: في حال وجود علاقة بين الغرض المستورد وغرض آخر موجود في قاعدة البيانات العاملة، مثل العلاقة بين الطالب والكلية، أو بين الكلية والجامعة، وغيرها.

5- التكرار: يمكن التحقق أيضاً من وجود تكرار للسجلات، إما ضمن الجدول العامل، أو ضمن الأغراض من الجدول الحالي، والتي تمتلك نفس المعطيات السامية للجدول العامل.

تسمح هذه الطريقة أيضاً بالتحقق من الأخطاء الدلالية، وذلك لأن الجدول مغلف ديناميكياً بمعطيات سامية تجعل لغة البرمجة المشتقة من النماذج MDP تراه وكأنه من نفس نمط الصف الحقيقي، ومن ثم سنتمكن من استدعاء أية عمليات تحقق ذات طابع دلالي موجودة مسبقاً في النظام. وأخيراً، وتحسبا لوجود قواعد عمل أو متنصتات لها قرار بصحة المعطيات المستوردة يمكن إجراءُ تجربةِ استيرادٍ ضمن مناقلة transaction تنتهي دوماً بالفشل، سواءً أَحَدَثَ خطأ دلالي أم لا. سيقوم النظام بإفشال المناقلة بعد أن تنتهي، لأن الهدف منها هو فقط التحقق من القدرة على

الاستيراد وضمان اكتشاف وجود أية متنصتات لها دور في التحقق من صحة الاستيراد.

في حال وجود أخطاء دلالية أو مفرداتية، تسمح المنظومة للمستخدم بتصحيحها مباشرةً دون إعادة استيراد، كما تتيح له في حال وجود أخطاءٍ متكررةٍ تصحيحها دفعة واحدة

لمعالجة احتمال تكرار نفس الغرض بأسماء مختلفة، تسمح المنظومة أيضاً باختبار وجود تشابهات في الأسماء من خلال إجراء عمليات عنقدة [20] لها، وفي حال تبين للمستخدم نتيجة العنقدة وجود تكرار أي كلمات متطابقة دلاليا ومختلفة إملائيا، سيتمكن من اختيار الاسم الصحيح واستبدال جميع التكرارات به. بعد الانتهاء من التحقق من صحة الجدول المستورد وتنظيفه، سيتمكن المستخدم من القيام بعملية استيراد نهائية.

4-2 القدرة على التراجع عن استيراد سجلٍ صحيح أو خاطئ

قد يجري اكتشافُ أخطاءٍ دلاليةٍ بعد مرحلة الاستيراد النهائي، وعندها تظهر الحاجة إلى حذف غرضٍ بعد استيراده فعلياً. كما قد يصبح السجل المستورد معطيات مرجعية لسجلاتٍ جرى استيرادها لاحقاً، كأن يجري استيراد رقم هاتف شركةٍ ما، ثم تنشأ لاحقاً سجلاتٍ تشير إلى رقم الهاتف المستورَد، وبالتالي، فإن حذف السجل "رقم هاتف" سيؤثر على سلامة السجلات التي تشير عليه.

من جهة أخرى، فإن التراجع إلى نسخةٍ قديمةٍ من قاعدة البيانات سوف يفقدنا استيراداتٍ أو أعمالٍ يوميةٍ قام بها آخرون، أو إعدادات أو غيرها. لذلك، فقد طورنا نماذج تصميمية تسمح لنا بالتراجع عن نتائج عملية الاستيراد دون التأثير على بقية المعطيات في قاعدة المعطيات وهي الحذف الشلالي Cascade Delete ونموذج تصميم الرصيد Balance Design Pattern.

أولاً، الحذف الشلالي: يسمح هذا النموذج المقترح بحذف الغرض وجميع الأغراض التي تشير إليه، وذلك بالاعتماد على البرمجة المشتقة من النماذج MDP. يمكن لهذا النموذج سحب معطيات تم استيرادها في لحظة معينة كما جرى استيراد معطياتٍ قبلها وبعدها، كما أنه يتيح حذف تأثير عملية استيراد معينة على المنظومة البرمجية. يجدر الانتباه هنا، إلى أن الحذف الشلالي لا يشبه الحذف التي تقدمه قواعد البيانات، والذي يقضي بحذف الغرض وجميع الأغراض التي يشير إليها بعلاقات تركيب composition، وإنما يجري حذف الغرض والأغراض التي تشير إليه وبشكل عودي. على سبيل المثال تقوم قواعد البيانات بحذف مكونات الغرض بشكل شلالي عند حذفه كأن تحذف المحرك عند حذف السيارة فأما في نموذجنا فإننا وعند حذف رقم هاتف ما فإننا نقوم بحذف جميع الإشارات إليه والموجود على هواتف الأصدقاء والزملاء. الهدف هو تأمين إمكانية التراجع من معطيات حية ربما استخدمها من استخدمها وبالتالي نضمن عند الحذف، وبعد تنبيه المستخجم لما سينتج عنه، أننا لم نحذف الغرض فقط وإنما جميع الإشارات إلى هذا الغرض. يعمل نموذج الحذف الشلالي كما يلي:

- إجراء مسح كامل للمعطيات السامية لجميع الأنماط للوقوف على نمط يشير نظرياً إلى نمط الغرض المراد حذفه.
- من أجل كل نمط يشير لهذا الغرض، يجري البحث عن الأغراض من هذا النمط والتي تشير فعلياً إلى الغرض المراد حذفه.

- من أجل كل غرض يشير للغرض المراد حذفه، يجري حذفه فعلياً، ولكن بشكل عودي باستعمال الحذف الشلالي.

في الواقع، لا يكفي الحذف الشلالي لمسح كل ما يتعلق بالغرض المراد حذفه، فهو يضمن حذف الأغراض التي نشأت بعد استيراده، لكن من الممكن أن يؤثر الغرض على قيم أغراضٍ موجودةٍ قبل استيراده، كما هو حالة الأرصدة Balance. فعند استيراد أصلٍ ثابتٍ مثلاً، سيرفع رصيد الشركة أو مستودع ما من هذا الأصل بمقدار واحد وهذه القيمة لن يتم التراجع عنها. جرى ابتكار نموذجٍ تصميمي خاصٍ هو نموذج تصميم الرصيد لحل هذه المسألة.

ثانياً، نموذج تصميم الرصيد: عادة ما يتم حساب الأرصدة، مثل عدد المواد في مستودعٍ، أو قيمة هذه المواد، وعدد المواد في الشركة، وقيمتها، أو حساب زبونٍ أو مزودٍ من خلال تحديث قيمة الرصيد بشكل تزايدي عند كل عملية إدخال أو إخراج. من سيئات هذه الطريقة، صعوبةُ تصحيحِ الأخطاءِ –إن وجدت– ويكمن الحل البديل في إعادة حساب الأرصدة عند الحاجة إليها، مما يعطي نتيجةً صحيحةً ومطابقة للواقع، إلا أن هذا الحل بطيء التنفيذ في حال تكرار الحاجة لحساب الرصيد. لذلك، طورنا نموذج تصميم الرصيد الذي يعتمد على حساب أكثر من رصيدٍ للهدف الواحد، وبشكل تزايدي، مما يعطي ميزات الطريقة التزايدية والطريقة الحسابية معاً. في الواقع، يجري حساب الأرصدة كما يلي:

- عند إنشاء أية عملية تؤثر على رصيد ما، يجري إنشاء نسخة محدثة من الرصيد بتاريخ العملية وبالشروط المطلوبة ولا يجري تحديث الرصيد السابق له.
- يجري حساب قيمة الرصيد الجديد من خلال البحث عن أقرب رصيد سابق له، وإضافة العمليات التي تمت بين تاريخ الرصيد السابق والرصيد الحالي.
- عند البحث عن رصيدٍ في تاريخ ما، يجري البحث عن أقرب رصيدٍ سابقٍ للتاريخ المطلوب، وتكون هي نفسها قيمة الرصيد في التاريخ المطلوب لأنه لا يوجد عمليات بين التاريخين. ثم نقوم حفظ هذا الرصيد الجديد في التاريخ المطلوب للاستفادة منه لاحقا.
- في حال التراجع عن عملية ما بتاريخ ما، أو إضافة عمليةٍ بمفعول رجعي، يجري حذفُ جميع الأرصدة منذ هذا التاريخ حتى اليوم ويعاد حسابها آليا كما في المرحلة السابقة.

من أهم ميزات هذه الطريقة أنها لا تستهلك حجوم تخزين ضخمة، لأنها لا تخزن رصيداً لكل يوم ولا لكل ساعة، وإنما حسب الطلب، ولا تحتاج إلى الكثير من الحسابات، لأنها تحسب من أقرب رصيدٍ، كما تسمح بتصحيح الأخطاء –إن وجدت– بحذف الأرصدة المتأثرة.

4-3 القدرة على إعادة بناء الروابط

دائما يوجد علاقات بين الأغراض وبالتالي بين الجداول في قاعدة البيانات ويجري بناء هذه العلاقات بناءا على روابط من نوع مفتاح أساسي ومفتاح اجنبي. يمكن بناء هذه العلاقات بطرق عديدة ولن تتطابق بالضرورة بين المنظومة المصدر والمنظومة الهدف. على سبيل المثال ربما كانت العلاقة بين الابن والأب مبنية على مفتاح أولي pk في النظام القديم، وعلى الرقم الوطني في النظام الجديد. سنقوم باستيراد الـ pk وfk والرقم الوطني لكل من الابن والأب إلى المنظومة الجديدة وإلى الجدول المؤقت، ومن ثم نقوم ببناء العلاقة الجديدة والمعروفة لدينا (لأننا نمتلك الروابط القديمة) وذلك عند نقل المعطيات من الجدول المتعدد الأشكال إلى الجدول العامل. إلا أننا سنبنيها وفقا للرقم الوطني لأننا لن ننقل pk or fk إلى الجدول الجديد لأنه لا مكان لهم فيه. وهنا، نجد فائدة إضافية للجدول متعدد الأشكال، حيث أنه يستطيع ديناميكياً أن يحاكي اجتماع

الجدول المصدر والجدول الهدف لأن عدد الأعمدة وأنماط الاعمدة التي يدعمها غير محدود وبالتالي نستطيع استيراد حقول مؤقتة إضافية لا مكان لها في الجدول الهدف.

4-4 القدرة على محاكاة الاستيراد

مهما كانت التحضيرات محكمة يوجد دوما مفاجآات غير متوقعة عند التنفيذ ضمن بيئة حية وقد تصادفنا أخطاء غير متوقعة ناجمة عن وجود قيود في البيئة الحية ليست متوفرة لدينا ولذلك فقد تضمنت منهجيتنا آلية لاستيراد تجريبي في البيئة الحية ينتهي دوما بالتراجع الحتمي وإعادة قاعدة البيانات لما كانت عليه سواءَ تكلل الاستيراد بالنجاح أو الفشل وذلك لإعطاء المستخدم الفرصة للحكم على نتيجة الاستيراد وقراءة الرسائل التي قد تظهر أثناء الاستيراد.

5 مثال عن إعدادات الجدول متعدد الاشكال

نجد أدناه مثالا عن إعدادات لجدول متعدد الأشكال

النمط المعني	نمط الأغراض التي ننتظر وصولها
شروط السلامة على المجموعة	قيود على مستوى المجموعة المستوردة كاملاً فيما بين الأغراض
شروط السلامة لكل غرض	
آلية الاستيراد النهائي	كيف يتم إدخال الأغراض السليمة والمدققة إلى الجدول الهدف في المنظومة الحية؟ ما هو الإجراء وما هي الخطوة؟
إلغاء استيراد	كيف يمكن التراجع عن إدخال غرض إلى الجدول الهدف في المنظومة الحية.
الحقول غير قابلة للتكرار	قيمته هي الحقول غير قابلة للتكرار ضمن الكتلة المستوردة
الحقول	الحقول المراد استيرادها
وصف الحقول	وصفٌ للحقول المراد استيرادها يتضمن أسماءها، وأنماطها، وقيمها المحتملة، وشروط السلامة الخاصة بها
الطباقMapping	العلاقة بين الحقول المستوردة والحقول الحقيقية الموجودة في الغرض الهدف

6 وصف المخرج النهائي

نقدم للمستخدم النهائي إجراءات عملٍ لاستيراد وتنظيف البيانات باعتماد المعطيات السامية الموجودة ضمن Meta Data ضمن البرمجية المشتقة من النماذج يعمل وفقاً للمراحل التالية:

1- تعريفُ معطياتٍ ساميةٍ تسمح برؤية الجدول متعدد الأشكال على أنه من نمطٍ مشابه للمصدر أو للهدف أو لاجتماعهما وإعطائها اسماً محدداً.
2- تحضير ملفات الاستيراد، وذلك بتصديرها من النظام الجديد بكل بساطة عند اختيار المعطيات السامية الموافقة.
3- ملء ملفات الاستيراد من خلال استيرادها من البرمجيات المصدر.
4- استيراد الملفات بصيغتها الخام إلى الجدول المؤقت، ويمكن للجدول المؤقت احتواء عددٍ كبيرٍ من المعطيات المختلفة من حيث النوع، كونه يخزن نوع الغرض ضمن كل سطر.
5- التحقق من صحة البيانات الإفرادية باستعمال check integrity، وإضافة أي شرط سلامة بفضل لغة ETDL المفسرة.
6- التحقق من عدم وجود تكرار في المجموعة الواحدة باستعمال العنقدة المعتمدة على المسافة المفرداتية المحرفية.
7- التحقق من قيود السلامة على مستوى إجمالي المجموعة المستوردة.
8- بناء حقول الربط مع المعطيات المرجعية، وفي حال عدم توفرها، اقتراح ربط مناسب للمستخدم اعتمادا على المسافة المفرداتية المحرفية.
9- تصحيح البيانات بشكل فردي أو تعميم التصحيح الفردي على كافة عناصر المجموعة.

10- اختبار الاستيراد الحقيقي النهائي
11- تطبيق الاستيراد الفعلي
12- حالات خاصة

a. التراجع عن الاستيراد إلى الجدول متعدد الأشكال باستعمال عمليات الحذف العادية.
b. التراجع عن الاستيراد النهائي بفضل الحذف الشلالي ونموذج تصميم الرصيد

7 الاختبارات

جرى اختبار المنظومة ضمن مشروع حساب الكلف للمشافي التابعة لوزارة الصحة ومنها مشفى المجتهد وهو أكبرها، ومشفى العيون، ومن ثم تابَعَ فريقُ وزارة الصحة بنفسه استيراد وتنظيف بيانات بقية المشافي منفرداً.

يعتمد حساب الكلف على مدخلاتٍ متعددةٍ تأتي من أقسامٍ مختلفةٍ في المشفى، وهي:

1- المشافي، وتأتي من وزارة الصحة.
2- الأقسام وهرميتها، وتأتي من الموارد البشرية ومن المستودعات ومن المحاسبة.
3- لائحة الموظفين، وتأتي من الموارد البشرية.
4- الرواتب، وتأتي من المحاسبة.
5- أصناف الاجهزة الطبية، وتأتي من القسم الهندسي.
6- أنواع الأثاث والاستهلاكات واللوازم الطبية، وتأتي من المستودعات.
7- أنواع الخدمات الطبية، وتأتي من الأقسام والمالية.
8- استلام المواد وتسليمها إلى الأقسام، وتأتي من المستودعات.
9- الأصول الثابتة، وتأتي من المحاسبة.
10- مبيعات الخدمات، وفواتير المرضى، وتأتي من الأقسام.
11- أسماء الموظفين، وتأتي من المستودعات ومن القسم الهندسي

تكمن المشاكل التي واجهناها في اختلاف أسماء الخدمات والأقسام والموظفين الواردة من مصادر مختلفة (قسم الأشعة هو نفسه قسم الصور الشعاعية وهو نفسه قسم التصوير) إضافة إلى وجود تكرار ونقص في المعطيات. أدى ذلك لاختلافات كبيرة في حساب الكلف فالنقص يؤدي إلى نقص في الكلف والتكرارا يؤدي إلى قسمة الكلف والواردات لقسم واحد بين قسمين وهمين ما هما إلا نفس القسم بإملاء مختلف.

في المرحلة الأولى، تم تطوير واختبار المنهجية الجديدة في مشفى المجتهد واستغرق العمل 9 رجل/شهر لتصدير المعطيات من مصادرها واستيرادها للمنظومة الجديدة واختبار المنظومة.

في المرحلة الثانية، جرى تطبيق المنظومة المطورة بنجاح في مشفى الأطفال، وبحيث تطلب العمل أياما معدودة حوالي 2 رجل/ يوم .

في المرحلة الثالثة، قام فريق من وزارة الصحة، وبدون أي تدخل من طرف المهندسين باستيراد وتنظيف البيانات لثمانية مشافي أخرى.

8 خلاصة

قمنا في هذا البحث، بطرح وتحقيق أسلوبٍ جديدٍ وفعالٍ لاستيراد وتنظيف البيانات أثناء التشغيل الحي، وذلك اعتماداً على البرمجة المشتقة من النماذج MDP، والمعطيات السامية. وتم تحقيق ذلك ضمن منظومة فعلية تسمح للمستخدم النهائي، الخبير في منطق العمل، باستيراد وتنظيف بياناته بنفسه، دون الحاجة لأية مهارات برمجية. كذلك، تسمح هذه المنظومة للمستخدم النهائي بالتراجع عن أخطاء الاستيراد، من خلال التراجع عن أثر عملية استيرادٍ محددةٍ، دون أن يتداخل مع عمل الآخرين، مما يسمح له بالإقلاع في العمل الحي مبكراً، ودون انتظار الانتهاء من استيراد كامل المعطيات التأسيسية والتحقق من صحتها.

لبناء هذه المنظومة جرى إنشاء 3 نماذج تصميمية معتمدة على المعطيات السامية، وهي الجدول متعدد الأشكال، الحذف الشلالي، نموذج تصميم الرصيد.

قمنا بتحقيق واختبار هذه المنظومة باستعمال بيئة تطوير مشتقة من النماذج هي بيئة الإكسير والتي تعتمد على بنية تحتية مفتوحة المصدر من J2EE و JBoss و MySQL.

جرى اختبار المنظومة والتحقق منها ضمن مشروع إدارة كلف الخدمات الصحية بالتعاون مع وزارة الصحة وفقا لثلاث مراحل الأولى مرحلة الاختبار الأول في مشفى المجتهد واحتاجت لثلاثة أشهر، ومن ثم مرحلة التحقق validation على مشفى العيون واحتاجت ليومين، ثم قامت وزارة الصحة باستعماله بشكل مستقلٍ تماماً على 8 مشافي إضافية.

9 مسرد المصطلحات

PIM (platform independent model)	نموذج مستقل عن المنصة
MDA (Model Driven Architecture)	البنيان المشتق من النماذج
MDP (Model Driven Programming)	البرمجة المشتقة من النماذج
EAI (Enterprise Application Integration)	مكاملة التطبيقات المؤسساتية
ETL (Extract Transform Load)	استخلاص وتحويل وتحميل
Enterprise Applications	التطبيقات المؤسساتية
Declarative Languages	اللغات التصريحية
3Tiers	ثلاثي الرتل
AOP (Aspect Oriented Programming)	البرمجة الموجهة بالسمات
MVC (Model View Controller)	نموذج وإظهار وتحكم
SOA (Service Oriented Architecture)	البنيان الموجه بالخدمات
Tier	رتل ويقصد به منظومة جزئية تعمل ضمن process خاص بها
layer	طبقة أي مجتزأ برمجي يشكل جزء من مجتزأ أكبر
Consistency	انسجام
interpretation	تفسير

References المراجع

[1] Zhang, X., Sun, W., Wang, W., Feng, Y., & Shi, B. (2006). Generating incremental ETL processes automatically. In First International Multi-Symposiums on Computer and Computational Sciences (IMSCCS'06) (Vol. 2, pp. 516-521). IEEE.

[2] Khayyat, Z., Ilyas, I. F., Jindal, A., Madden, S., Ouzzani, M., Papotti, P., & Yin, S. (2015). Bigdansing: A system for big data cleansing. In Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data (pp. 1215-1230). ACM.

[3] Lv, Z., Deng, W., Zhang, Z., Guo, N., & Yan, G. (2019). A Data Fusion and Data Cleaning System for Smart Grids Big Data. In 2019 IEEE Intl Conf on Parallel & Distributed Processing with Applications, Big Data & Cloud Computing, Sustainable Computing & Communications, Social Computing & Networking (ISPA/ BDCloud /SocialCom/SustainCom) (pp. 802-807). IEEE.

[4] Diao, Y., Liu, K. Y., Meng, X., Ye, X., & He, K. (2015). A big data online cleaning algorithm based on dynamic outlier detection. In 2015 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (pp. 230-234). IEEE.

[5] Yu, H., Xiao-yi, Z., Zhen, Y., & Guo-quan, J. (2009). A universal data cleaning framework based on user model. In 2009 ISECS International Colloquium on Computing, Communication, Control, and Management (Vol. 2, pp. 200-202). IEEE.

[6] Bhattacharjee, A. K., Chatterjee, P., Shaw, M. P., & Chakraborty, M. (2014). ETL based cleaning on database. International Journal of Computer Applications, 105(8).

[7] BIRST. Comparing the Total Cost of Ownership of Business Intelligence Solutions [White Generating Incremental ETL Processes Automatically Paper]. Retrieved from www.whitepapers.em360tech.com.

[8] Wang, H., & Ye, Z. (2010). An ETL Services Framework Based on Metadata. In 2010 2nd International Workshop on Intelligent Systems and Applications (pp. 1-4). IEEE.

[9] Bertossi, L., Kolahi, S., & Lakshmanan, L. V. (2013). Data cleaning and query answering with matching dependencies and matching functions. Theory of Computing Systems, 52(3), 441-482.

[10] Koshley, D. K., & Halder, R. (2015). Data cleaning: An abstraction-based approach. In 2015 International Conference on Advances in Computing, Communications and Informatics (ICACCI) (pp. 713-719). IEEE.

[11] Bouzeghoub, M., Fabret, F., & Matulovic-Broqué, M. (1999). Modeling the Data Warehouse Refreshment Process as a Workflow Application. In the Proceedings of the International Workshop on Design and Management of Data Warehouse, DMDW (Vol. 19, p. 6).

[12] Kent, S. (2002, May). Model driven engineering. In International Conference on Integrated Formal Methods (pp. 286-298). Springer, Berlin, Heidelberg.

[13] "The Essence of Model Driven Architecture", Wim Bast, www.jaxmagazine.com/ itr/ online_artikel/ psecom,id,548,nodeid,147.html

[14] Joukhadar, A. (2008). "EliXir: a framework for Building e-business applications." Information and Communication Technologies: From Theory to Applications. ICTTA 2008. IEEE.

[15] Mellor, S. J., Scott, K., Uhl, A., & Weise, D. (2004). MDA distilled: principles of model-driven architecture. Addison-Wesley Professional.

[16] Ghafour, H. H. A., El-Bastawissy, A., & Heggazy, A. F. A. (2011, November). AEDA: Arabic edit distance algorithm towards a new approach for Arabic name matching. In Computer Engineering & Systems (ICCES), 2011 International Conference on (pp. 307-311). IEEE.

[17] Haldar, R., & Mukhopadhyay, D. (2011). Levenshtein distance technique in dictionary lookup methods: An improved approach. arXiv preprint arXiv:1101.1232.

[18] Gueddah, H., Yousfi, A., & Belkasmi, M. (2015). The filtered combination of the weighted edit distance and the Jaro-Winkler distance to improve spellchecking Arabic texts. In Computer Systems and Applications (AICCSA), 2015 IEEE/ACS 12th International Conference of (pp. 1-6). IEEE.

[19] Kaushik, M., & Mathur, M. B. (2014). Comparative study of K-means and hierarchical clustering techniques. International Journal of Software & Hardware Research in Engineering (IJSHRE), 2(6).

[20] Tan, P. N., Steinbach, M., & Kumar, V. (2013). Data mining cluster analysis: basic concepts and algorithms. Introduction to data mining, 487-533.

[21] Joukhadar A. (2020). Run Then Specify: An Intelligent Framework For Building E-Business Applications. International Journal of Scientific & Technology Research (09), 117-122.

Received	**2020/11/4**	**إيداع البحث**
Accepted for Publ.	**2021/2/17**	**قبول البحث للنشر**