IL 4GL DEL 21° SECOLO

dal monitor a fosfori verdi allo smartphone

Genero®

All’apice della sua gloria, il 4GL (IBM® Informix® 4GL) ha dominato i settori del retail, della finanza, dell’industria e della pubblica amministrazione grazie alla sua capacità di creare le migliori applicazioni aziendali in termini di affidabilità, velocità e costi contenuti.

A metà degli anni Novanta tale dominio è stato rallentato dall’avvento del client-server computing e dei linguaggi di programmazione orientati agli oggetti, anche se quelle tecnologie non hanno mai potuto eguagliare il 4GL in quanto a produttività degli sviluppatori.

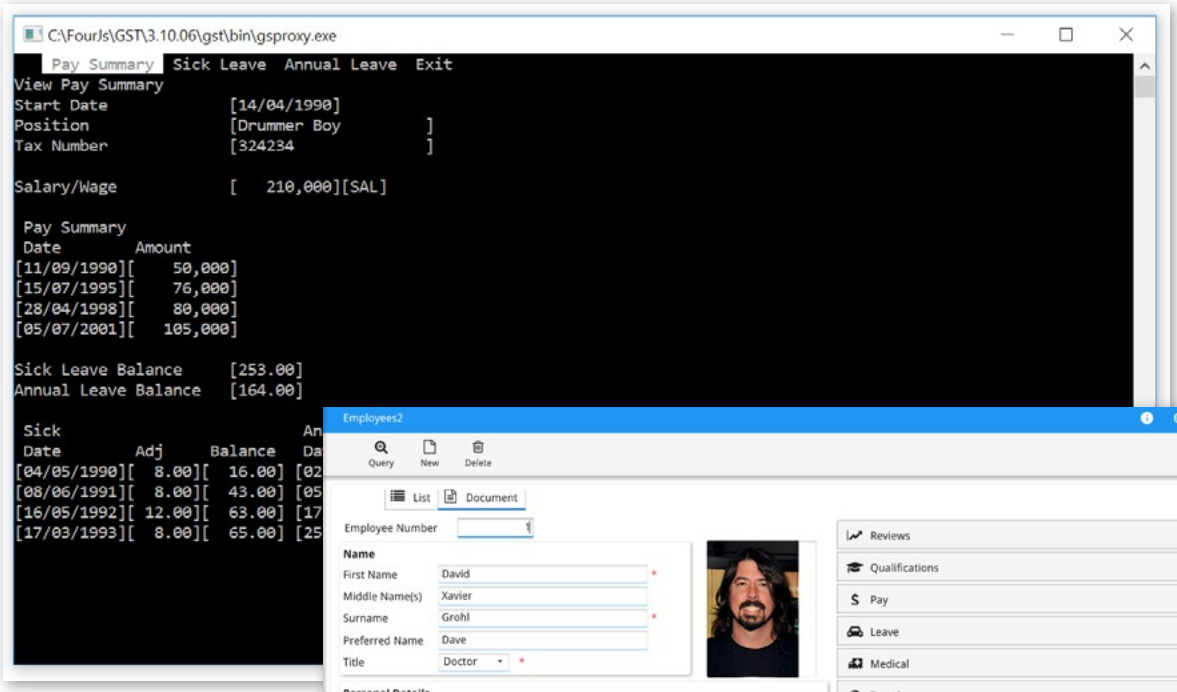
In un mondo in cui i budget dell’ IT sono costantemente sotto pressione, il 4GL rimane a tutt’oggi una tecnologia rilevante e pervasiva, ed è ancora il modo più produttivo per sviluppare applicazioni aziendali.

Negli ultimi 30 anni, i progressi tecnologici hanno rivoluzionato il nostro modo di lavorare. Durante quel periodo, il 4GL ha faticato a tenere il passo, motivo per cui è stato creato Genero®.

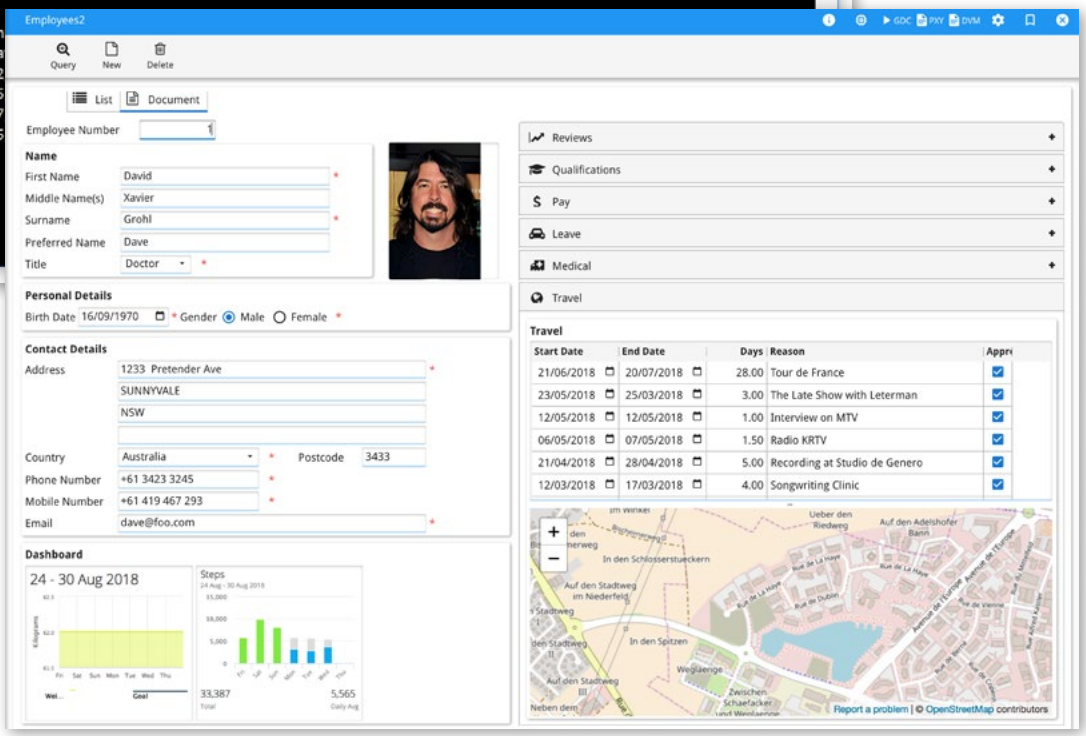
Genero è stato scritto da zero utilizzando le più recenti tecniche e tecnologie, pur rimanendo compatibile al 100% con il 4GL. Genero consente alle aziende di conservare gli importanti investimenti in scrittura di codice e in competenze fatti negli ultimi decenni. È uno degli ambienti di sviluppo più portabili, estensibili e scalabili che consente alle applicazioni di evolversi senza sforzo al passo con gli ultimi trend tecnologici.

In ambiente Genero, le applicazioni 4GL possono essere eseguite ovunque, indipendentemente dal dispositivo client, dal database o dal sistema operativo con poche o nessuna modifica. Sia su un monitor a fosfori verdi che su un laptop macOS® o MS® Windows, in un browser o su uno smartphone, le applicazioni si adattano in fase di runtime da un unico codice sorgente. Questo spiega perché Genero continua a promuovere la causa 4GL e a prosperare come elemento chiave di tanti sistemi informativi mission-critical di grandi aziende e istituzioni governative.

> Genero protegge decenni di investimenti in codice e competenze.



4GL



Genero

Il 4GL non si è mai distaccato dalla sue radici di "schermo a fosfori verdi" degli anni '80 e non ha mai fornito ai clienti una user experience contemporanea. Di conseguenza, non supporta nessuna delle funzionalità grafiche ed ergonomiche disponibili sui moderni client: MS Windows, macOS, browser o dispositivi mobili.

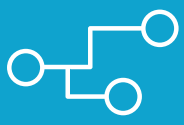
In assenza di tab, widget e barre degli strumenti, le applicazioni sviluppate con 4GL tendono a concentrare quante più informazioni possibili in un singolo modulo diviso in finestre «principale» e «dettaglio».

Genero pone fine a tutto questo, rivoluzionando la userexperience e rendendo le applicazioni indistinguibili dal resto.

> Genero è unico nella sua capacità di combinare il codice scritto 25 anni fa con un nuovo codice che sfrutta i più recenti paradigmi informatici.

ERGONOMIA CONTEMPORANEA

In 4GL, le finestre di dialogo 'pop-up' forniscono un meccanismo per accedere a più informazioni, ma gli utenti devono uscire da una finestra prima di entrare in un'altra. Gli utenti possono «saltare» da campo all'altro, ma solo sequenzialmente e in una finestra alla volta. Le applicazioni 4GL sono poco fruibili tramite browser, dove è facile perdere il contesto tramite il pulsante «indietro».
Di conseguenza, sono macchinose e frustranti da usare per chi è abituato con i sistemi moderni. Genero risolve questi problemi con una tecnologia di client-rendering all'avanguardia.

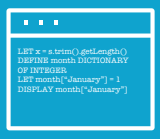


INTEROPERABILITÀ

Il 4GL è notoriamente un linguaggio "introverso"; Parla molto bene a se stesso e ignora tutti gli altri. Non è che non voglia comunicare - semplicemente non sa come farlo. Le applicazioni moderne, indipendentemente dal modo in cui sono state scritte, interagiscono tramite web services SOAP e RESTful utilizzando strutture dati XML e JSON. Queste funzionalità sono state incorporate in Genero per rendere le applicazioni aperte, in grado di comunicare senza problemi con il mondo esterno e di essere utilizzate senza la necessità di una conoscenza approfondita dei protocolli sottostanti.

SICUREZZA

Le applicazioni esposte al mondo esterno sono tuttavia estremamente vulnerabili agli attacchi. E' quindi fondamentale sviluppare applicazioni sicure che si integrino perfettamente in questo mondo complesso. Genero fornisce classi e metodi che eseguono crittografia, algoritmi di codifica, autenticazione, certificazione e single sign-on in modo che le app possano interagire con i social media e altri sistemi orientati a Internet.



MIGLIORAMENTI LINUSTICI

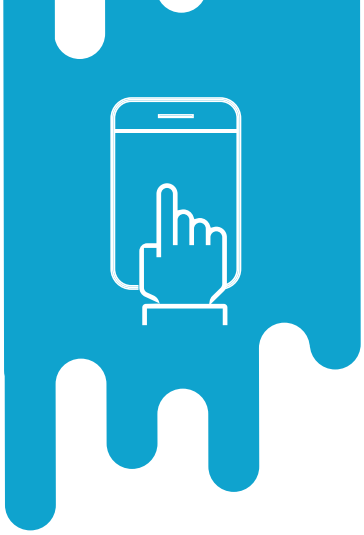
Il linguaggio è stato migliorato per incorporare funzionalità come i parametri nominati, il passaggio dei parametri funzione per riferimento, la gestione degli eventi, l'internazionalizzazione e l'interfacciamento a Java®. Una classe di metodi astratti di sistema operativo per la gestione del filesystem migliora notevolmente la portabilità delle applicazioni.

UN AMBIENTE DI SVILUPPO INTEGRATO

Ogni linguaggio di sviluppo moderno ha bisogno di un ambiente ricco all'interno del quale esprimereappieno le proprie potenzialità. Genero Studio, Genero Report Writer e Genero Ghost Client servono a questo scopo: migliorare la produttività degli sviluppatori e testare le applicazioni per validità, affidabilità e scalabilità.

ARCHITETTURA

Genero libera lo sviluppatore dai vincoli del 4GL – nel caso specifico come il suo essere mono-database, mono-sistema operativo, monolitico e a "fosfori verdi". Liberando lo sviluppatore da questi limiti, Genero infonde nuova linfa vitale nelle applicazioni legacy preparandole per l'ampio spettro di implementazioni dell' oggi e del domani. Lo fa all'interno di un unico ambiente di sviluppo, creando applicazioni portabili +, sicure e scalabili a migliaia di utenti simultanei senza la necessità di costosi middleware.



ERGONOMIA CONTEMPORANEA

È possibile ricompilare l'applicazione 4GL con Genero 'as is' utilizzando la Text User Interface (TUI) e continuare a sviluppare come al solito in modalità 'ASCII'. In alternativa, la ricompilazione in «modalità tradizionale» genera «automagicamente» un'interfaccia utente grafica, tasti funzione soft, barre di scorrimento e un mouse senza che sia necessaria alcuna modifica al codice.
Questo approccio è veloce, ma l'interfaccia utente mantiene i vincoli della sua geometria ASCII. Questo sarebbe un peccato, perché i benefici della GUI (Graphical User Interface) di Genero non verrebbero sfruttati.

Forse la modifica più importante apportata al linguaggio in Genero è l'introduzione della parola chiave LAYOUT per i file .per. Quando viene utilizzato al posto di SCREEN, la form viene generata tramite l'attivazione di una sintassi di sovrainsieme comprendente verbi grafici e token.

In molti casi, è possibile ottenere il massimo impatto visivo semplicemente modificando i file .per più importanti. Genero estende la sintassi della form .per con i seguenti elementi grafici astratti che eseguono il rendering in modo nativo su Windows, Linux®, macOS, browser e API mobili:

widget grafici definiscono come i valori vengono visualizzati e modificati in una form della GUI (Graphical User Interface):

- BUTTON: presenta una zona selezionabile o cliccabile sullo schermo che attiva un'azione.
- BUTTONEDIT: un edit con un pulsante associato utilizzato per attivare un'azione di modifica o fornire maggiori dettagli su un valore.
- CHECKBOX: l'utente può scegliere tra due (o tre compreso NULL) valori
- COMBOBOX: l'utente può selezionare da un elenco scorrevole di valori
- DATEEDIT: l'utente può selezionare una data da un widget di calendario
- DATETIMEEDIT: l'utente può selezionare una data e un'ora da un widget specializzato
- EDIT:rappresentazione e modifica di default di un valore
- IMAGE: generazione dell'immagine
- LABEL: visualizza il testo che non è modificabile, ad es. titoli, descrizioni
- PROGRESSBAR: visualizza un integer come una barra che aumenta fino al valore massimo
- RADIOGROUP: l'utente può selezionare un valore tra più opzioni
- SLIDER: l'utente può aumentare o diminuire un valore integer facendo scorrere un widget
- SPINEDIT: l'utente può regolare un valore integer selezionando i pulsanti su / giù
- TEXTEDIT: l'utente può modificare più righe di testo
- TIMEEDIT: l'utente può selezionare un'ora da un widget specializzato

contenitori di layout definiscono in che modo i widget della GUI sono disposti l'uno rispetto all'altro

- GRID - posiziona i widget figli con un dato parametro X, Y, larghezza e altezza
- SCROLLGRID: organizza più istanze di una GRID
- TABLE – organizza i widget figli in una o più righe identiche di una struttura simil tabellare
- TREE - simile a TABLE ma con una struttura ad albero collassabile e gerarchica
- STACK: consente di disporre i widget figli in uno stile nativo all'interfaccia utente
- GROUP: posiziona un bordo attorno a diversi contenuti secondari e facoltativamente assegna un titolo
- FOLDER + PAGES - organizza i contenitori figli in più pagine in cui è visibile solo il contenuto di una pagina e gli altri vengono visualizzati come schede o cartelle
- VBOX: impila i contenitori figli verticalmente
- HBOX: organizza i contenitori figli in orizzontale

Componenti Web: consente l'utilizzo di tecnologie HTML di terze parti esistenti (HTML / JS / CSS) e la creazione di componenti di interfaccia utente personalizzati. Esempi di utilizzo includono: embedding di video, documenti HTML, mappe, grafici,modificarich text, calendario, acquisizione firme, richtables, mappe a immagine, POS touch screen, autenticazione SSO. I componenti Web predefiniti forniscono funzionalità avanzate di modifica di testo, la selezione di più immagini, e il rendering SVG.

Action defaults: valori predefiniti di azione che centralizzano la definizione di testo, commenti, immagini, acceleratori e altre proprietà delle azioni.

Top menus (Menu bar): disposizione strutturata delle opzioni di menu nella parte superiore di una finestra.

Toolbars: disposizione di pulsanti lungo la parte superiore, inferiore, sinistra o destra del modulo.

Parallel dialogs: permet consente di dividere lo schermo con ogni riquadro che esegue il codice indipendentemente.

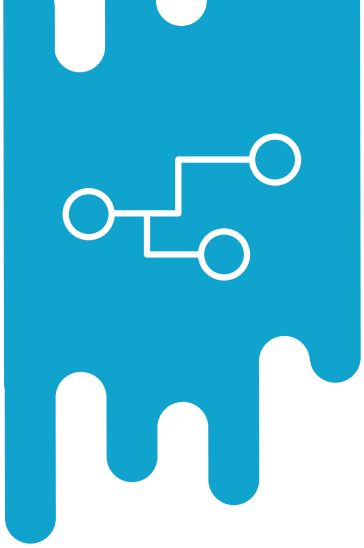
Multiple dialogs: unisce più istruzioni di interazione utente (MENU, INPUT, INPUT ARRAY, DISPLAY ARRAY, CONSTRUCT) in un'unica finestra di dialogo attiva che facilita la navigazione dell'utente.

Dynamic dialogs: construit un'istruzione di interazione utente (INPUT, CONSTRUCT, DISPLAY ARRAY, INPUT ARRAY, DIALOG, ...) definita in fase di esecuzione anziché in fase di compilazione. Ciò consente l'uso di codice generico con una struttura dati variabile. Utilizzato in modo ottimale in combinazione con la creazione dinamica di moduli in fase di esecuzione e utilizzando i cursori del database dinamico dalla classe base.Sqlhandle.

Multiple selects: possibilità di selezionare più righe all'interno di un array, allo scopo di eseguire un'azione come l'eliminazione o la stampa.

Focus on field: possibilità di selezionare una cella da un DISPLAY ARRAY, non solo una riga.

Drag and drop: consente di eseguire il "drag and drop" di determinati oggetti uno sopra l'altro per eseguire una determinata azione, all'interno o tra le applicazioni.



INTEROPERABILITÀ

Genero si è evoluto negli ultimi 25 anni per includere le ultime API e strutture dati per l'interazione con altri sistemi. Ciò significa che comprende le strutture dati più diffuse: XML e JSON così come i metodi più diffusi: Web Services, sia SOAP che RESTful.

Strutture dati

XML

Genero offre un set completo di classi e metodi per lavorare con i documenti XML. Questo include il supporto sia dei modelli DOM (Document Object Modeling) e StAX (Streaming API per XML) nonché delle serializzazioni e XSLT.

JSON

Genero fornisce classi per lavorare con JSON (JavaScript Object Notation) - un formato leggero di interscambio di dati molto noto.

Metodi

Web services – Genero fornisce le librerie per creare servizi Web, sia come server o provider del servizio Web, sia come client del servizio Web.

SOAP – una parte fondamentale di questo servizio è il file WSDL che fornisce i dettagli dell'interfaccia. Prendi una funzione 4GL esistente con parametri di input e output definiti e crea o pubblica un'operazione di servizio Web basata su tale funzione.

RESTful – più leggero di SOAP. Passa messaggi (JSON, XML in genere) in un formato a vostra scelta.

File – L'uso di queste classi elimina completamente la necessità di librerie «C».

Base.Channel – prima dell'esistenza dei servizi Web, è possibile che vi siate interfacciati ad altri sistemi lettura/scrittura su file. Poiché l'API di chiamata di sistema di 4GL era così povera, dovevate scrivere librerie scritte in C e/o script di shell. La classe base.Channel è un insieme di metodi utilizzati per leggere e scrivere su file e può essere utilizzata per sostituire C e script di shell.

OS – questa classe consente di elencare i file nelle directory e testare l'esistenza dei file e le autorizzazioni. Ciò elimina le dipendenze dal sistema operativo e rende il codice portabile.

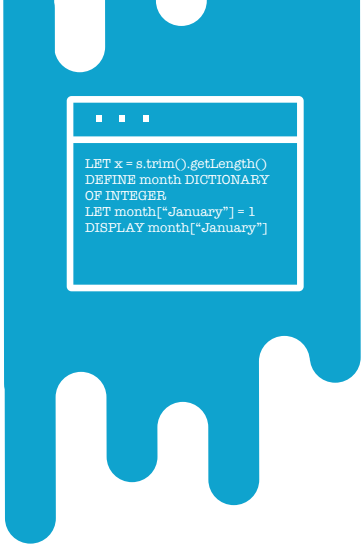
Sockets – sono stati aggiunti alla classe base.Channel metodi per facilitare la comunicazione via socket, sia come client che come server a / da un determinato server e porta.

Pipes – il metodo openPipe è stato aggiunto alla classe base.Channel per facilitare la comunicazione con un sotto-processo tramite un canale pipe.

SICUREZZA

Security Class: il pacchetto di sicurezza fornisce classi e metodi per eseguire la crittografia di base. Sono comprese classi PBKDF2 e BCrypt per l'archiviazione e il controllo di password crittografate. Sono inoltre fornite le classi per implementare algoritmi di codifica come SHA512, MD5 (ecc.) così come classi per gli algoritmi HexBinary e Base64.

Secure Communications: se un'applicazione vuole scambiare messaggi con un'applicazione finanziaria, aziendale o personale sul Web, deve essere in grado di autenticare l'origine del messaggio, assicurarsi che nessuna applicazione dannosa abbia alterato il messaggio originale e assicurarsi che nessuna applicazione esterna possa intercettare il messaggio. Ciò coinvolge certificati, autorità di certificazione, chiavi private, ecc. tutti disponibili in Genero.



MIGLIORAMENTI LINGUISTICI

La sintassi del linguaggio 4GL è stata notevolmente migliorata eliminandone i limiti e conformandola alle attuali abitudini di programmazione. Ecco un elenco (esemplificativo ma non esaustivo) di miglioramenti che abbiamo apportato:

Sintassi

keywords – STRING, CONSTANT, BOOLEAN, TYPE, BIGINT, TINYINT, PUBLIC, PRIVATE, NVL, IFF, SFMT, DICTIONARY, DYNAMIC ARRAY, TRY/CATCH

ON ACTION – Il blocco ON ACTION esegue il codice quando si verifica un evento utente:

```
ON ACTION zoom
  piuttosto che:
ON KEY(F5)
```

L'azione «zoom» è specificata da un formato .per o da un file action default ed eseguita indipendentemente dal modo in cui è stata attivata.

Passare i parametri funzione per riferimento – Questa funzionalità è utile quando si ha a che fare con strutture complesse, record e array. Le strutture complesse vengono passate alla funzione senza la necessità di gestire il risultato di ritorno. Solo una sua istanza si trova in memoria, evitando due copie (una nella funzione chiamante e una nella funzione chiamata), consumando così meno memoria.

```
DEFINE arr DYNAMIC ARRAY OF complexDataType

# arr contiene dati non trasformati
CALL transform_array(arr)
# arr contiene dati trasformati
```

Inizializzazione letterale – I letterali possono essere inizializzati semplicemente:

```
TYPE colorType RECORD
   red, green, blue INTEGER
END RECORD

DEFINE purple colorType = (255,0,255)
```

Case sensitivity - Se selezionato, questa diventa sintassi non valida:

```
DEFINE foo, Foo, FOO STRING
```

e:

```
DEFINE accountCode STRING
DEFINE accountCodeLength INTEGER
        LET accountCodeLength = accountCode.getlength()
```

Parametri nominati – I parametri nominati rimuovono la necessità di specificare ogni argomento e ottenere il loro ordine quando chiamano le funzioni. I parametri non specificati sono impostati su valori predefiniti. Ciò migliora la leggibilità del codice.

```
CALL fgl_report_configurePDFDevice(fromPage = 1, toPage = 1)
```

Funzionalità di codice varie – *Questi sono esempi di sintassi visti in altri linguaggi, ma non in 4GL. Ciò facilita la migrazione per gli sviluppatori che sono poco avvezzi al 4GL.*

Typedeclaration – sono consentite nei parametri di chiamata e restituiscono tipi di funzioni: una modalità abbreviata per migliorare la sicurezza dei tipi:

```
FUNCTION add(x INTEGER, y INTEGER) RETURNS INTEGER
```

Abbreviazione del concatenamento di metodi degli oggetti

```
LET x = s.trim().getLength()
```

Un nuovo tipo di dati DIZIONARIO (noto anche come tabella hash) utilizza una stringa come chiave per le tabelle di indicizzazione. Può memorizzare tutti i tipi di dati Genero:

```
DEFINE month DICTIONARY OF INTEGER
    LET month["January"] = 1
    DISPLAY month["January"]
```

ANYRECORD *– tipo di record generico*
Al momento della compilazione, è noto solo il tipo di una funzione (parametri, valori restituiti) quando si utilizzano i riferimenti alle funzioni. Il tipo di record generico ANYRECORD viene utilizzato quando una funzione accetta come input un record la cui struttura è sconosciuta.

IMPORT – Se esiste una libreria esterna che si desidera utilizzare all'interno del proprio codice, è possibile utilizzare la dichiarazione IMPORT come segue:

- IMPORT FGL - per importare una libreria 4GL
- IMPORT - per importare una libreria «C»
- IMPORT JAVA - per importare una libreria Java

La nostra comunità di sviluppatori usa ad esempio le dichiarazioni IMPORT JAVA per i seguenti scopi:

- Creazione di file MS® Word ed Excel utilizzando la libreria POI di Apache
- Caricamento di video su You-Tube®

Internazionalizzazione - Le applicazioni Genero sono progettate per funzionare in tutto il mondo tramite il supporto per UTF-8 e il codice indipendente dal set di caratteri.

Una *stringa localizzata* è un testo dell'applicazione definito nei cataloghi di stringhe al di fuori del codice sorgente standard. Ciò facilita la manutenzione del testo tradotto e l'uso di una particolare traduzione in fase di esecuzione senza la necessità di ricompilare.

Un'*applicazione locale* determina quale localizzazione è in uso per un determinato utente client. Pertanto, la stessa istanza dell'applicazione può supportare più lingue contemporaneamente. Per il mondo arabo è supportata anche la lettura da destra a sinistra e la semantica di lunghezza BYTE e CHARACTER è supportata per l'Asia.

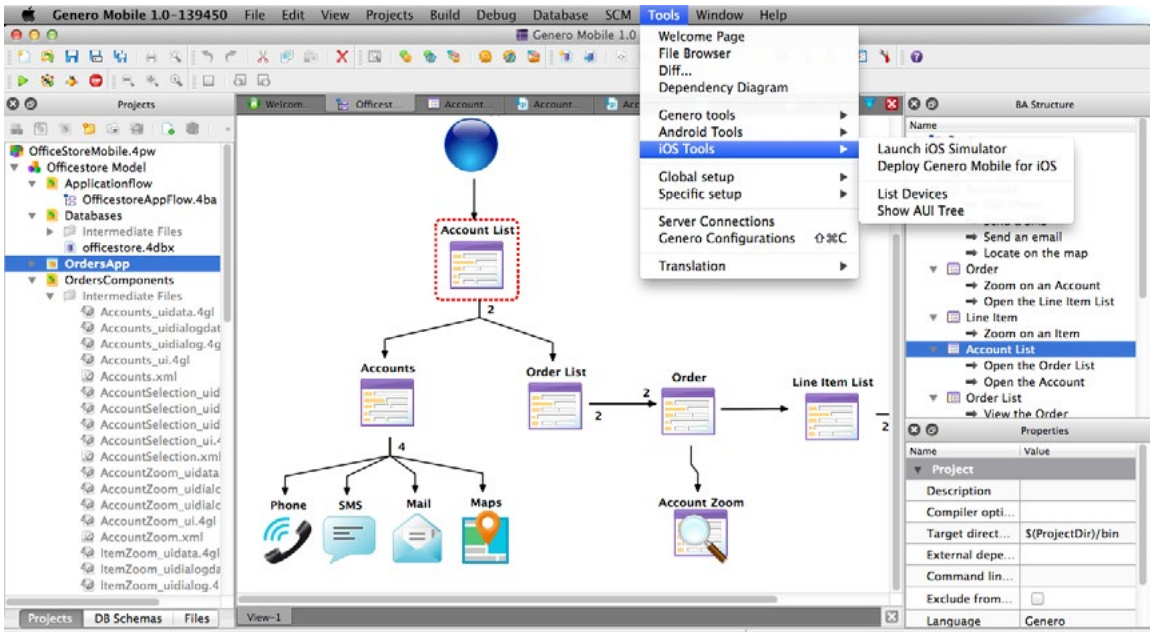
UN AMBIENTE DI SVILUPPO INTEGRATO (GENERO STUDIO)

Genero Studio è un ambiente di sviluppo integrato (IDE) progettato per migliorare la produttività degli sviluppatori. Consente il debug simultaneo di applicazioni sviluppate per desktop, browser e dispositivi mobili.

Include un editor di codice language-sensitive che supporta il completamento del codice, il controllo della sintassi in tempo reale, l'evidenziazione delle parole chiave 4GL e altri formati di file. Include anche un debugger grafico, un form-designer grafico , un report writer grafico, un project manager per la creazione di applicazioni complesse, un generatore di schemi di database, un meta-schema, un generatore di codice, uno strumento di modellazione visiva, strumenti di analisi e profilazione.

UNA PIATTAFORMA DI SVILUPPO LOW-CODE

Business Application Modeller (BAM) di Genero Studio è una piattaforma di sviluppo low-code (LCDP) destinata agli sviluppatori che cercano un ciclo di vita incrementale, continuo e multi piattaforma che utilizza diagrammi anziché codifica manuale.



BAM - integrazione di app native per smartphone in un modulo Genero Mobile

Il codice per applicazioni e servizi Web è definito in template e modellato visivamente, il che garantisce coerenza e codice strutturato organizzati in livelli:

- Storage dei dati
- Servizi dati
- Pubblicazione di servizi seb (RESTful, SOAP)
- Forms e reports

La maggior parte dei modelli di programmazione specifici per il business sono trattati nei modelli predefiniti come moduli «CRUD», elenchi, dettagli, filtri, navigazione, gestione della concorrenza e reports. Le app mobili native come contatti, calendario e telefono possono essere integrate tramite la definizione nel modello. Ciò consente di descrivere ampie porzioni dell'applicazione come diagrammi senza dover scrivere alcuna linea di codice. I diagrammi possono poi essere modificati dallo sviluppatore per migliorare o alterare il comportamento dell'applicazione. Le regole di business su misura possono essere codificate negli eventi.

POTENTI AIUTI ALLO SVILUPPO

Questi strumenti possono essere attivati in fase di compilazione o di esecuzione da Genero Studio o da riga di comando.

Profiler – Profiler identifica i colli di bottiglia del programma. Evidenzia la percentuale di tempo impiegato nelle funzioni e il numero di volte in cui le funzioni vengono richiamate da altre funzioni.

Code coverage – Lo strumento di coverage del codice sorgente analizza se un processo di test sta testando ogni riga di codice in un'applicazione. Se eseguite un programma più volte con la funzionalità di code coverage abilitata, verrà tenuto un contatore per ogni singola riga di codice mentre viene eseguita.

```
35  1 MAIN
36
37        DEFINE
38            r_output STRING,
39            r_filename STRING
40
41    1   LET r_filename="OrderReport"
42    1   LET r_output="Genero Report Viewer"
43    1   OPEN FORM f_configuration FROM "Configuration"
44    1   DISPLAY FORM f_configuration
45    1   INPUT BY NAME r_filename, r_output  ATTRIBUTES (UNBUFFERED,
46            BEFORE INPUT
47    1           CALL dialog.setActionActive("saveondisk",r_output !=
48            ON ACTION preview
49    1           CALL runReport(r_filename, r_output,"preview")
50            ON ACTION saveOnDisk
51    0           CALL runReport(r_filename, r_output,"saveOnDisk")
52            ON ACTION quit
53    1           EXIT INPUT
54            ON CHANGE r_output
55    0           CALL dialog.setActionActive("preview",r_output != "I
```

Trace – Trace è una utile funzionalità di debug che crea un registro completo o filtrato delle chiamate che sono state fatte da un programma, quali erano gli argomenti e cosa è stato prodotto.

Shared libraries – L'architettura di deployment si basa su librerie condivise, che eliminano la necessità di costruire dei «runners».

Strumenti per la rientranza e riformattazione

TEST AUTOMATICI (GENERO GHOST CLIENT)

Il client Genero Ghost offre le seguenti funzionalità:

- Effettua test unitari utilizzando scenari che convalidanola business logic
- Esegue i test di carico per determinare il numero di utenti che l'infrastruttura può supportare
- Effettua comparazionidei tempi di risposta a carichi elevati

UN GENERATORE DI REPORT (GENERO REPORT WRITER)

L'istruzione REPORT in 4GL viene utilizzata per produrre report di gestione, che per gli standard odierni appaiono molto arretrati.

Genero Report Writer fornisce un sofisticato sistema di pubblicazione di report per la produzione di report aziendali di alto livello, fatture e etichette di spedizione per citarne solo alcuni usi. La sua peculiarità consiste nella sua capacità di trasmettere grandi volumi di documenti a stampanti, display o addirittura a diversi formati di file, tra cui: pdf, svg, xlsx, docx, html, ps, jpg e png. Lo streaming consente la stampa immediata della prima pagina quando ci sono centinaia o addirittura migliaia di pagine da stampare. Risparmia preziose risorse di sistema per documenti voluminosi eliminando la necessità di creare file temporanei di grandi dimensioni.

Genero Report Writer supporta gli oggetti grafici più comuni come: immagini, tabelle, codici a barre, codici QR e grafici, oltre a gestire diversi tipi di caratteri tipografici, punti tipografici e colori. I layout sono dinamici per facilitare la trasformazione dei documenti da un fattore di forma all'altro, senza la necessità di rielaborare il progetto.

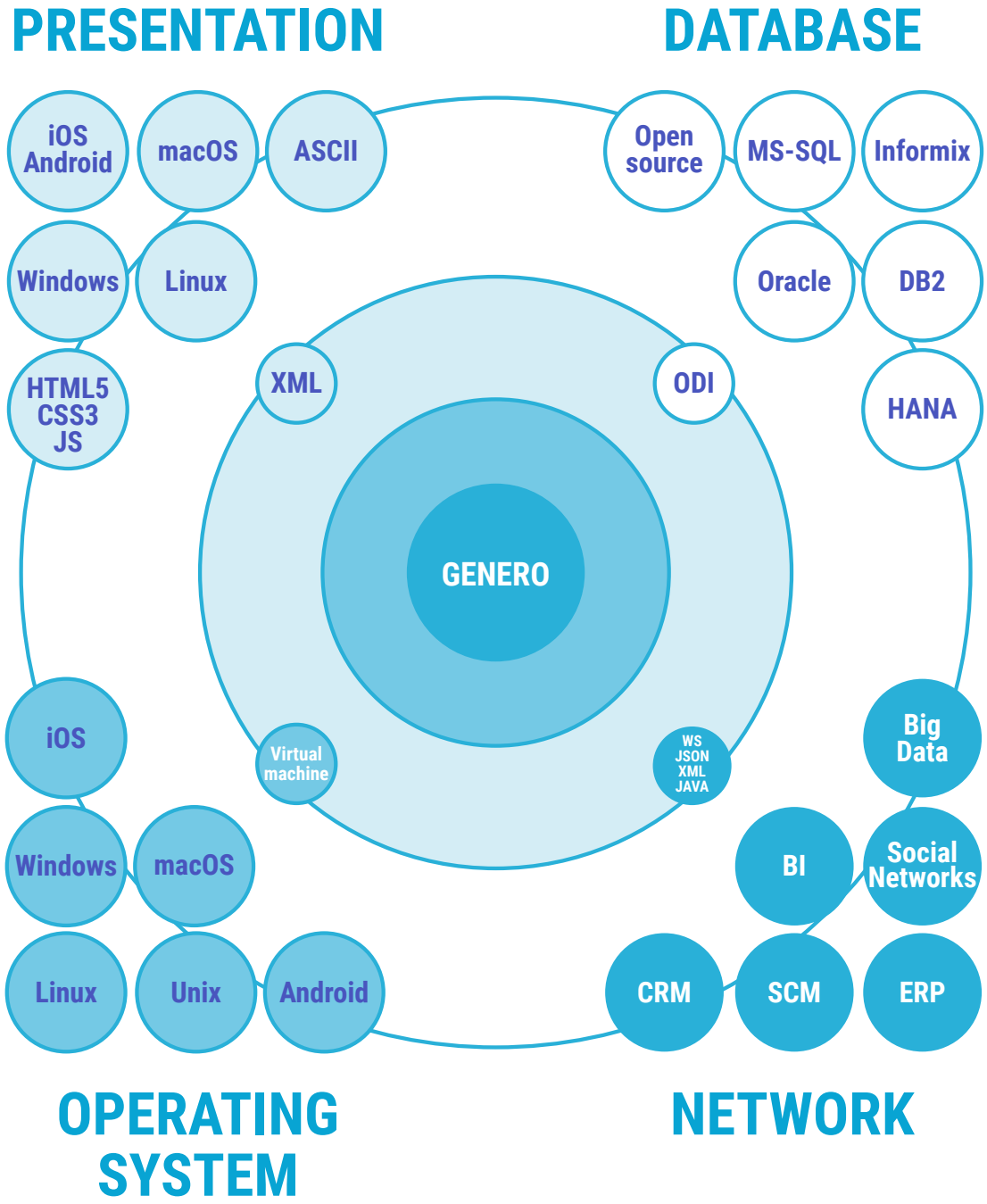
I report vengono creati con il Graphical Report Designer, un quasi WYSIWYG designer che consente di modificare semplicemente i formati di pagina senza dover rivedere il layout del report.

Genero Report Writer può essere utilizzato con linguaggi diversi da Genero come Java, «C» e PHP.

Modalità di compatibilità

La Modalità di compatibilità consente di generare i report 4GL esistenti utilizzando uno dei metodi supportati da Genero Report Writer.

ARCHITECTURE DE GENERO



L'architettura di Genero è sicura, portatile e scalabile per migliaia di utenti ed integra il suo middleware evitando così spese aggiuntive inutile

WORLDWIDE OFFICES

USA & CANADA
Four Js Development Tools Inc.

251 O'Connor Ridge Blvd. Suite 125
Irving, Texas, 75038
United States

Toll free phone: 866 314 7300 (Free)
Phone: (972) 893-7300
Fax: (972) 893-7304

LATIN AMERICA
Four Js Development Tools Latin America SACV.

Insurgentes Sur 1602
Col. Credito Constructor
Mexico City, DF 03940
Mexico

Phone: +52 (55) 1000 9160

EUROPE
Four Js Development Tools Europe Ltd.

Suite 5, Rineanna House
Shannon Free Zone
Shannon V14 CA36
Co Clare, Ireland

Phone: +353 61 708 314
Fax: +353 61 708 315

ASIA
Four J's Development Tools Asia

Suite 210 29 Kiora Rd,
Miranda NSW, 2228 Sydney
Australia

Phone: +612-8004-5890

FRANCE
Four Js Development Tools SARL.

28 Quai Gallieni
92 150 Suresnes
France

Phone: +33 1 41 38 86 30
Fax: +33 1 41 38 84 46

UNITED KINGDOM
Four Js Development Tools UK LTD.

Regus House, Victory Way Admirals Park
Dartford, DA2 6QD
United Kingdom

Phone: +44 (0) 370 111 5140
Fax: +44 (0) 370 111 5154

CENTRAL EUROPE
Four Js Development Tools Software Vertriebs GmbH

Leonhardsweg 2,
82008 Unterhaching
Germany

Phone: +49 89 60815400
Fax: +49 89 60815402

IBERICA
Four Js Development Tools Iberica LTDA.

Martinez Villergas 49, 1a Planta,
28027 Madrid
Spain

Phone: +34 91 047 76 61
Fax: +34 91 047 76 57

ITALIE
Four Js Development Tools Italia

Via Ferruccio Ferrari,
6 42124 Reggio Emilia
Italie

Phone: +39 (0)522 1712600
Fax: +39 (0)522 1712940

www.4js.com

Trademarks

® Four Js, Genero and its logos are registered trademarks of Four J's Development Tools Europe Ltd.
® Mac , macOS and IOS are registered trademarks of Apple Corps.
® IBM, Informix and DB2 are registered trademarks of IBM Corporation.
® Microsoft, MS Windows, and MS SQL are registered trademarks of Microsoft Corporation.
® Java and Oracle are registered trademarks of Oracle Corporation.
® Linux is a registered trademark owned by Linus Torvalds.
® SAP HANA is a registered trademark of SAP SE in Germany and other countries.
® UNIX is a registered trademark of The Open Group for the United States and other countries.
® You Tube and Android are registered trademarks of Alphabet Corp.

© 1995-2018 All rights reserved.

FOUR Js
The Power of Simplicity