XIII SIMPASE
SIMPÓSIO DE AUTOMAÇÃO DE SISTEMAS ELÉTRICOS

15 a 18 de setembro de 2019
Shopping RioMar
Recife, Pernambuco

AB/YY

# METODOLOGIA PARA AUTOMATIZAR A GERAÇÃO DE APLICAÇÕES SCADA EM SISTEMAS ELÉTRICOS

**Clovis Simões (*)**
**Spin Engenharia de Automação**

**Túlio Pereira da Silva**
**Spin Engenharia de Automação**

SUMÁRIO

Este trabalho apresenta uma metodologia utilizada no desenvolvimento de aplicações de software tipo SCADA, orientadas ao controle de um processo qualquer, que reduz em muito o tempo de desenvolvimento e implantação da aplicação, assim como aumenta sua qualidade e minimiza a possibilidade de erros.
Essa metodologia, designada Lean Automation (LA), se fundamenta no conceito de componente, que passa a ser um objeto do SCADA, incluído na aplicação em um único clique, que contém toda uma funcionalidade orientada a monitoração e controle do processo ou parte dele. Com alguns cliques pode-se adicionar alguns componentes ao SCADA que correspondem a toda a aplicação
Esta metodologia permite implantar aplicações elétricas em tempo recorde (parametrização, TAF e TAC), com alta qualidade e nível de padronização, baixíssimo nível de erros e baixo custo. Além disso, mantém o conhecimento do processo de implantação no software e não só nos técnicos que participaram do empreendimento.

PALAVRAS-CHAVE

SCADA; Lean Automation; Geração automática de aplicação; Automação de subestações e Gestão de Ativos

## 1.0 INTRODUÇÃO

Um dos autores participou no desenvolvimento de quatro sistemas SCADA, no decorrer desses últimos 30 anos, e foi integrador de dezenas de aplicações utilizando esses sistemas desenvolvidos. Dessa experiência e da utilização de ferramentas de última geração no desenvolvimento do seu último SCADA, surgiu a metodologia que está incorporada ao próprio software, isto é, foram desenvolvidos módulos de software que automatizam o processo de geração de aplicações, a partir do conhecimento dos procedimentos de integração usados na automação de sistemas elétricos, bem como o conhecimento dos protocolos de comunicação utilizados e o aspecto regulatório associado ao processo elétrico controlado, a partir de normas estabelecidas pelas agências reguladoras.
O processo para a geração automática de todos os pontos de um sistema, telas, relatórios, etc., seja de automação de uma subestação, seja de automação de um parque eólico, baseia-se um conjunto de procedimentos que devem ser desenvolvidos de forma organizada, seguindo uma sequência, que nos leva, ao final, a atividade de parametrização do SCADA, testes em ambiente de laboratório, testes em ambiente de fábrica, com os cubículos elétricos recém montados, e comissionamento no campo após a instalação dos cubículos e passagem dos cabos.
Se considerarmos a aplicação de automação de uma subestação de distribuição de concessionaria de energia, a unidade a ser automatizada é o vão (bay). Uma subestação, basicamente, é composta por uns poucos tipos de vãos: o vão de linha, de transformador, de bancos de capacitores, de alimentação, de serviços auxiliares, etc. Esses vãos são tratados de forma padronizada pela concessionária. Assim, por exemplo, qualquer vão de uma subestação de distribuição, tem um modelo de dados que se repete, com informações do tipo: dados de estados de equipamentos de manobra, dados de medidas elétricas, dados de proteções dos equipamentos existentes no vão, dados de eventos e alarmes associados aos estados e medidas do vão, dados de regras para armazenar as informações em arquivos históricos, etc. Essas informações são colocadas em modelos de dados (templates) e, quando se define a arquitetura de uma nova subestação a ser implantada, rotinas, automaticamente, explodem todos os dados que serão tratados nesses modelos, gerando toda a base de dados da aplicação, com seus eventos, alarmes e arquivos históricos. Além disso, o próprio tipo do vão é associado ao seu desenho esquemático, seguindo regras associadas a cultura da empresa. Os desenhos desses vãos são colocados em bibliotecas de figuras e também instanciados, automaticamente.
Assim, a partir dos modelos associados a cultura da empresa, ao se informar os tipos de vãos de uma dada subestação e quantas unidades de cada vão existirão, é possível gerar, automaticamente, todos os pontos da subestação, com os alarmes / eventos associados, com seus endereços nos equipamentos de proteção existentes, com as tabelas com o desenho do unifilar geral e telas de detalhe, com os relatórios usualmente usados pela concessionária para a monitoração e controle da subestação.

(*) Granja do Torto - Parque Tecnológico de Brasília - BIOTIC, Lote 4, 1º andar – CEP 70635-815 Brasília-DF– Brasil Tel: (+55 61)99284-2034 – E-mail: simoes@spinengenharia.com.br.

O mesmo se aplica para parques eólicos, disponibilidade de ativos, subestações industriais, usinas, etc. Uma vez gerada uma aplicação base, cada funcionalidade é testada individualmente, assim como cada animação de tela, cada evento, cada alarme, etc. Após esses testes, gera-se novas aplicações muito rapidamente, sem erros.

## 2.0 TRABALHANDO COM COMPONENTES

### 2.1 Conceito de componentes

A célula básica da metodologia LA é designada componente. Assim, um projeto vazio de um software SCADA exige que se gaste algumas dezenas de horas até se concluir uma aplicação. Um projeto de Lean Automation, por default, já vem um componente inicial que cria a estrutura básica de um projeto de automação de sistemas elétricos. Este componente inicial estabelece um padrão de telas, cria alguns templates e tags que permitem gerar, automaticamente, vários relatórios para consulta de dados de tempo real e históricos, assim como, possuem janelas de comando usuais em sistemas de automação tais como abertura e fechamento de disjuntores / seccionadores, aumentar e diminuir tap de transformadores, etc. A figura 1, abaixo, permite visualizar o componente default de uma aplicação, designado Componente Spin, que contém parte de um projeto de automação com:

- Layout default das telas de processo, com um header de relatórios e teclas básicas de navegação, um corpo a ser adicionado os desenhos tipos unifilares e um rodapé com os últimos alarmes, data e horário;
- Dicionários em português e inglês de palavras usadas nos relatórios e nas mensagens de eventos e alarmes tipo aberto/fechado, normal/atuado, abrir/fechar, etc. Dessa forma o projeto poderá ser usado tanto em português como em inglês.
- Relatórios de: (1) Alarmes correntes; (2) Eventos do dia, (3) Log de operação, (4) Tags de variáveis do projeto que atendem a um filtro; (5) Eventos históricos que atendem a um filtro; (6) Medidas históricas que atendem a um filtro; (7) Tendências de tempo real ou históricas que atendem a um filtro; (8) Formulário de construção de filtros; (9) Janela para inserir outros relatórios construídos na aplicação; (10) Janela de exportação dos relatórios para Excel;
- Scripts para a carga inicial do sistema;
- Tags e templates usados nos relatórios básicos do sistema;

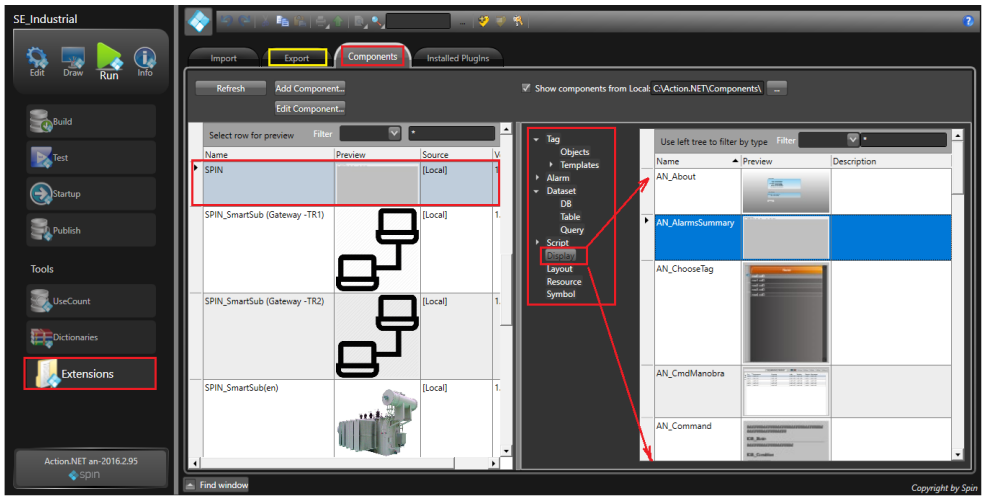

FIGURA 1 – Componente Default adicionado a um projeto vazio

- Criação da estrutura default hierárquica dos objetos;
- Definição de categorias que associam a objetos uma ou mais funcionalidades tais como vai ou não para histórico, calcula o desbalanço de corrente de circuitos de duas ou três fases, etc.;
- Segurança default do projeto com tipos de usuários, suas permissões e políticas aplicadas;
- Janelas de anotações associadas a equipamentos, janelas de manobras de equipamentos, etc.

Este componente básico dá uma estrutura comum a qualquer aplicação desenvolvida, podendo ser designado como parte da cultura da empresa, de sua metodologia de automação dos processos. Se o usuário desejar alterar este ou qualquer outro componente, basta ele adicioná-lo ao projeto, alterá-lo e exporta-lo, usando a aba com destaque amarelo na figura acima, criando um novo componente. Esta funcionalidade dá uma nova dimensão ao uso de componentes, isto é, a partir de um componente com dezenas de funcionalidades o usuário pode copiá-lo e modifica-lo, criando um novo componente ou mais adequado a sua cultura ou melhor que o componente anterior.

## 2.2 Componentes e melhores práticas

Um componente pode ser usado por um fabricante ou um consultor para fixar as melhores práticas de uso de um equipamento qualquer como, por exemplo, uma nova linha de relés. Assim, o desenvolvedor usa todas as funcionalidades disponíveis no equipamento, para tirar o máximo dele, e com isso fixa esse conhecimento no componente que deverá ser usado em dezenas de projetos, garantindo a propagação das melhores práticas para o uso daquele equipamento em novos projetos.
Considerando soluções de gestão de ativos elétricos como a monitoração de transformadores, hoje existem IEDs de diferentes fabricantes que utilizando uma rede descentralizada [1] fazem esta monitoração e muitos destes IEDs já são vendidos embarcados no transformador. Além disso, muitos relés usados para a proteção de transformadores, também acumulam funcionalidades de monitorar o ativo. Usando esta ideia básica, a título de exemplo, criou-se o componente "Asset Monitoring" que é gerado a partir da inclusão de vários componentes onde, cada um deles pode ser tanto o IED de um fabricante, conforme a figura abaixo, como um relé de proteção que acumula funcionalidades de gestão de ativo.

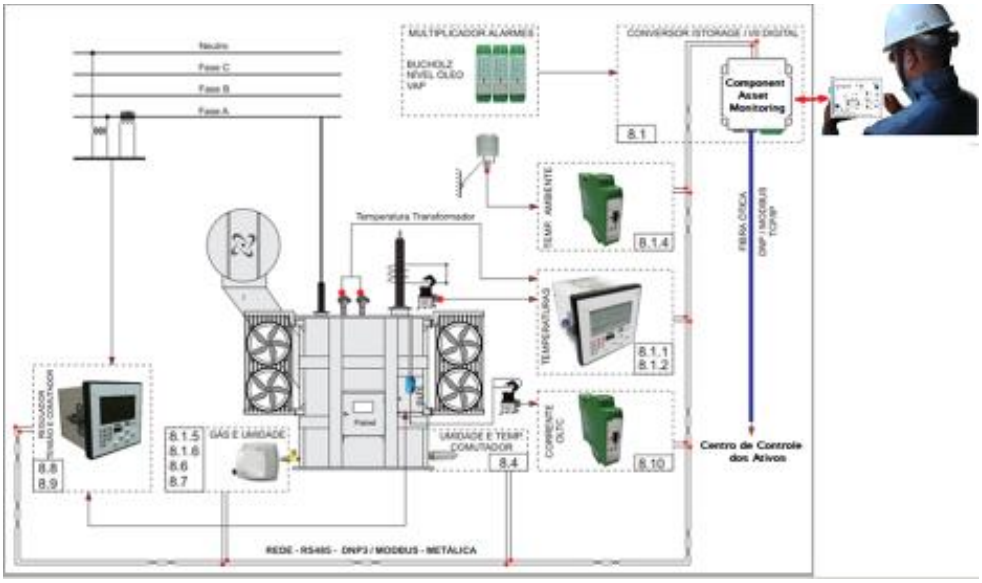

FIGURA 2 – IEDs de diversos fabricantes monitorando o transformador

Este componente é apenas um exemplo, mas a ideia proposta é que se crie uma biblioteca com todos os IEDs normalmente utilizados para a monitoração dos transformadores de uma concessionária e, em função dos IEDs existentes em uma dada subestação, em poucos minutos se gera a aplicação que coleta os dados desses IEDs e os apresente em telas locais, quando existe alguém presente na subestação, assim como os envie para o sistema central, onde será feita a monitoração de todos os ativos como é o caso de um sistema para gestão de ativos centralizado [2].
O uso de componentes no contexto de melhores práticas, além de garantir a criação de aplicações com qualidade, sem erros, em tempo recorde, permite que todo o conhecimento do melhor uso do IED aplicado à solução fique no componente e não na cabeça de técnicos.

## 2.3 Desenvolvendo aplicações com componentes

O desenvolvimento de um componente exige mais tempo que uma aplicação similar, pois você deve criar um conjunto de funcionalidades reusáveis e, para tal, deve definir uma arquitetura de solução um pouco mais complexa. Quanto aos testes de homologação do componente, estes são similares aos de uma aplicação, já que ambos devem garantir o perfeito funcionamento da solução, mas no caso de componentes, uma vez testado poderá ser reusado dezenas de vezes sem nova necessidade de teste. Dessa forma, em pouco tempo você será mais produtivo, mais assertivo, enfim, mais competitivo. Para exemplificar essa mudança comportamental, apresenta-se a seguir o desenvolvimento de uma nova aplicação solicitada por um cliente e como a visão de desenvolver através de componente mudou a perspectiva de solução da empresa. O cliente neste caso, possui nove sítios automatizados com este software, sendo cinco complexos eólicos e quatro usinas hidrelétricas e solicitou que se adicionasse às nove soluções relatórios de disponibilidade dos ativos elétricos e, para tal, foi criado um único componente, genérico, que analisa a disponibilidade de ativos elétricos. Este mesmo componente foi adicionado à nove aplicações distintas, gerando relatórios e dashboards que seguem um mesmo padrão. A figura abaixo mostra alguns dashboards e relatórios gerados pelo componente.

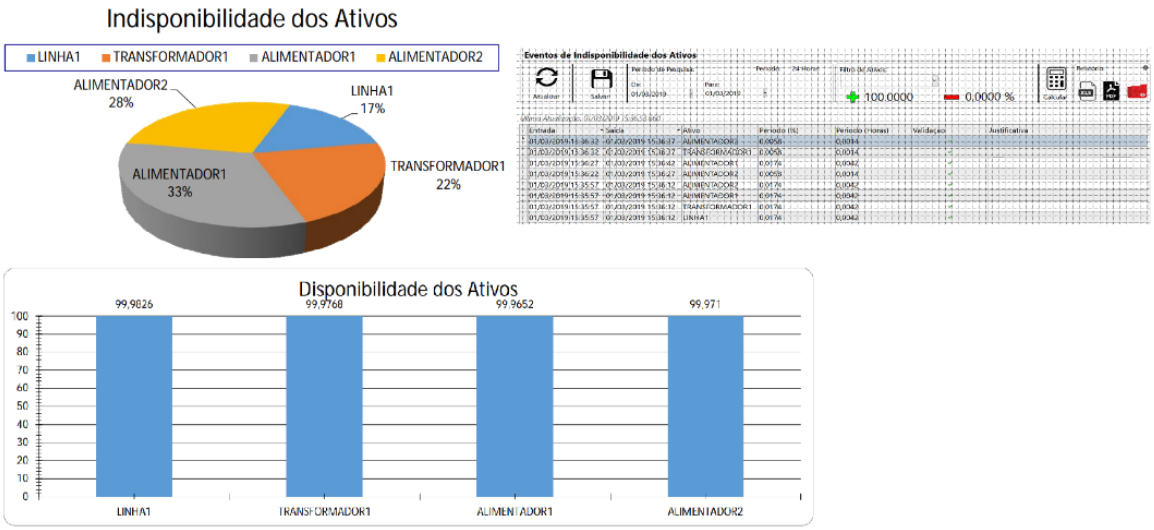


FIGURA 3 – Disponibilidade de ativos elétricos

A alteração e teste de uma aplicação existente, adicionando-se o relatório de disponibilidade de seus ativos, em todos os vãos, dura menos de quatro horas e os relatórios podem ser gerados por turno, dia, semana, mês e ano. Isto dá um ganho de produtividade substantivo, e mão de obra usada é de um estagiário, que faz ações mecânicas, a partir da documentação do componente.
Concluído o desenvolvimento desse componente, a empresa pode usá-lo em qualquer outro sítio elétrico (usina, subestação, parque eólico, etc.) para analisar a disponibilidade dos ativos, obtendo relatórios padronizados.

## 2.4 Geração automática de aplicações

Após esta apresentação do conceito e da forma de uso dos componentes, vamos ver como criar bibliotecas de componentes e como utilizá-los para, em poucas horas, gerar diferentes aplicações, com muita qualidade, robustez e baixo custo de desenvolvimento e implantação.
O projeto base, designado projeto default, já não é um projeto vazio, pois possui um primeiro componente que contém um conjunto de ferramentas básicas de uma aplicação elétrica, segundo a experiência dos autores.

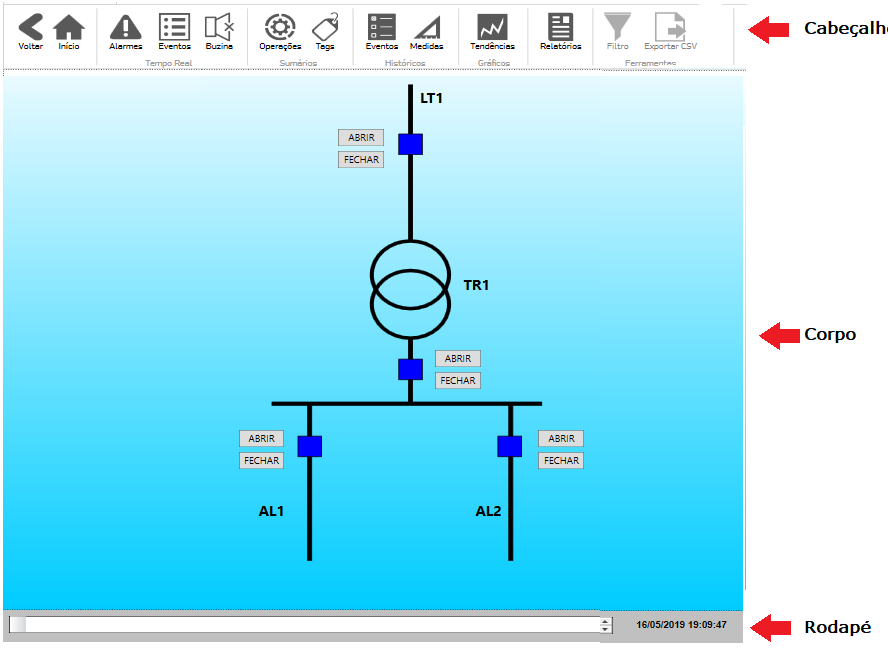


FIGURA 4 - Modelo das telas do projeto básico

Neste projeto temos o modelo de todas as telas, com um cabeçalho, um corpo e um rodapé onde:

- O cabeçalho tem regras básicas de navegação (retorne a tela anterior e tela principal), um conjunto de relatórios de tempo real e históricos, um filtro genérico aplicável a qualquer relatório, um botão de silenciar alarme sonoro e um botão para exportar um relatório selecionado;
- O corpo possui telas de processo ou o corpo de relatórios selecionados;
- O rodapé possui o último alarme e, se clicarmos sobre ele, aparecerão os últimos “x” alarmes. Possui também a data e hora corrente.

Além disso, existem janelas com todos os comandos possíveis, associados a cultura do usuário, anotações para que o operador possa comentar ações como, porque impediu o comando de um equipamento, regra de nomeação de tags, biblioteca com todos os tipos de símbolos de visualização utilizáveis na cultura do usuário (vãos tipo com todas as funcionalidades já disponíveis), tipos de alarme aplicáveis, regras para enviar eventos, alarmes e medidas para histórico, etc.
Observar que todos estes elementos são módulos chamados por referência, isto é, se um usuário clica sobre o símbolo de um disjuntor, para executar um comando, o nome do tag do disjuntor é enviado por referência para a janela de comando e seu template é, automaticamente, mapeado para aquele dispositivo, isto é, a janela de comando serve para todos os equipamentos daquele tipo, assim como as telas de detalhes disponíveis. Por exemplo, se eu tenho uma tela de detalhe de um alimentador, esta tela é única para todos os alimentadores e quando se navega para ela, o alimentador selecionado é passado por referência transformando a tela na tela detalhe daquele alimentador. Abaixo, a título de exemplo, uma tela detalhe de um alimentador de um componente que contém a cultura de automação de subestações de uma concessionária.

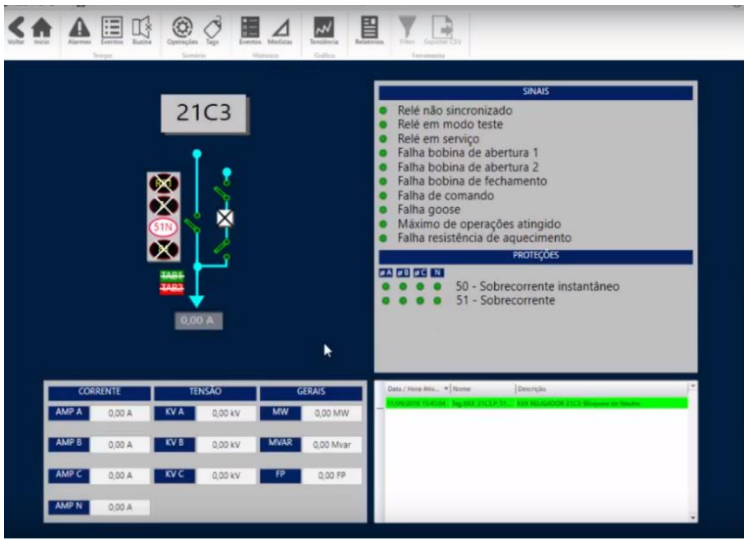


FIGURA 5 – Tela detalhe de um alimentador segundo o padrão de uma concessionária

Como o projeto default é um componente, é possível modifica-lo para adequar-se à cultura de diferentes concessionárias.
Sobre um projeto default, adiciona-se outros componentes que possibilitam criar diferentes aplicações SCADA como, por exemplo, uma subestação de concessionária, um complexo eólico, um módulo de disponibilidade de ativos elétricos, etc. A partir deste conceito, o próprio software já está preparado para importar componentes de uma biblioteca, isto é, quando instalado é criado um diretório que deve ser usado como um repositório de componentes, como mostrado na figura abaixo.

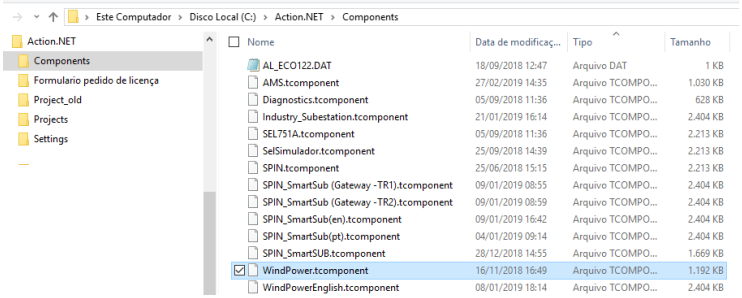


FIGURA 6 – Repositório de Componentes

Usuários credenciados podem acessar uma biblioteca de componentes na nuvem e fazer o download de um ou mais componentes e de sua documentação, bem como pode fazer o upload de componentes desenvolvidos por ele, com a respectiva documentação.
Para preservar o direito autoral de componentes sofisticados, é possível gera-los como “PlugIns”, que são componentes não editáveis, isto é, você pode usá-los em sua aplicação, mas não pode ver seu código nem o modificar.
Assim, com o uso de componente e da metodologia LA, muda-se o paradigma de desenvolver aplicações SCADA. Uma aplicação poderá ser criada em minutos e com o tempo os usuários produzirão aplicações extremamente sofisticadas com a garantia de qualidade, robustez e zero erros.

## 3.0 TRABALHANDO COM BIBLIOTECAS EXTERNAS E COMPONENTES

Com o uso de componentes, mostramos como é possível criar uma fábrica de software para geração de aplicações elétricas de sistemas SCADA com baixo custo, alta qualidade e sem erros.
O próximo passo em direção a soluções robustas, que reduzam o custo de produção com garantia de qualidade é adição de bibliotecas externas sejam elas “open source” ou compradas. O ambiente DotNET permite se adicionar bibliotecas externas (DLLs), como exemplifica a figura abaixo, onde foram adicionadas três bibliotecas externas “open source”, a primeira que permite utilizar mapas georreferenciados de diferentes fabricantes, a segunda que permite desenvolver telas tridimensionais e a terceira, que é a biblioteca do EPRI, designada *Open Distribution System Simulator* (OpenDSS). Essa última é um programa *de* simulação para sistemas elétricos de distribuição de energia elétrica. O OpenDSS é capaz de realizar novos tipos de análises que são necessárias para atender as necessidades futuras relacionadas as Redes Elétricas Inteligentes (*Smart Grids*) e muitos dos seus recursos foram, originalmente, desenvolvidos para dar suporte às necessidades das análises em que há geração distribuída (GD).
Essas bibliotecas, entretanto, são um conjunto de código, propriedades, atributos e métodos que devem ser usados, adequadamente, para serem úteis na configuração do SCADA. Por exemplo, o mapa GIS apresentado abaixo foi usado em um cliente para mapear as chaves de poste do sistema distribuição, e para tal deve ser possível em tempo de configuração inserir chaves georreferenciadas no mapa, com propriedades que permitam informar, em tempo real, seu estado, suas principais medidas, traçar as redes elétricas entre elas, bem como criar métodos que deverão comandá-las.
No caso de bibliotecas externas o componente vem como resposta a forma de integrar bibliotecas externas ao SCADA, isto é, o componente é a parte do projeto que não só inclui o a biblioteca externa ao sistema como também manipula suas propriedades e métodos no contexto do SCADA, tornando-o um objeto parametrizável por qualquer cliente que deseja utilizá-lo.

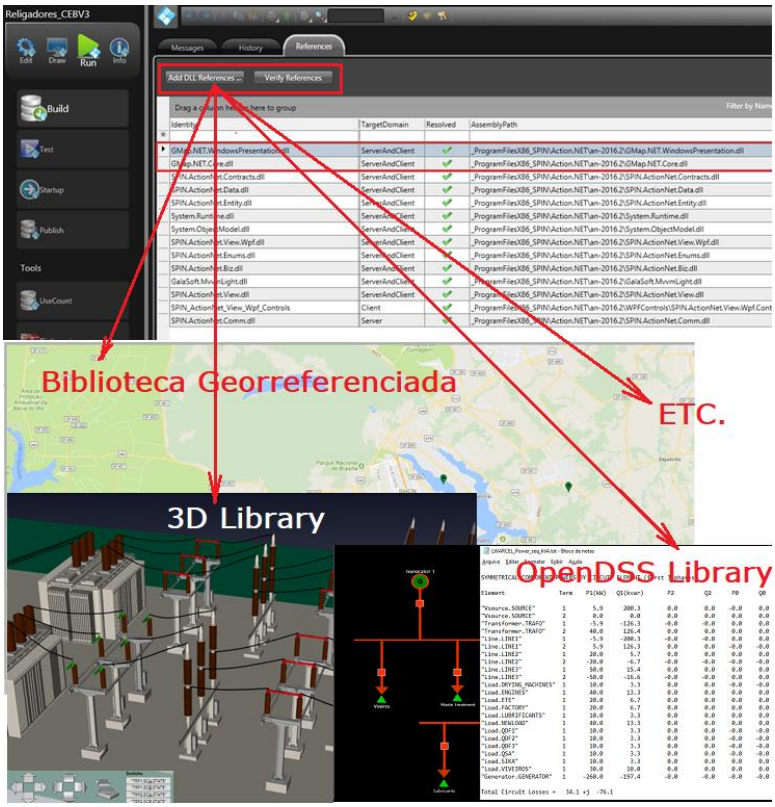


FIGURA 7 – Inserindo bibliotecas externas através de componentes

Dando continuidade ao exposto no capítulo anterior, através da metodologia, expande-se as funcionalidades do software com as novas bibliotecas que já vem prontas para serem usadas na forma de componente e se no futuro o usuário decidir usar mais funcionalidades da nova biblioteca, basta alterar o componente para tal. Da mesma forma, se novas versão da biblioteca forem lançadas, basta incorporar as novas funcionalidades ao componente existente.
A integração de bibliotecas Open Source ao software SCADA pode ser um atalho em direção ao Smart Grid [3].

## 4.0 USANDO EXTENSÕES

Finalmente, uma última ferramenta para automatizar a geração de soluções de tempo real, é o uso de extensões, que são módulos de software desenvolvidos em C# ou VB.NET, usados em tempo de parametrização para automatizar a geração da base de dados de tempo real e histórica, assim como facilitar a parametrização da aplicação. Abaixo são mostrados alguns exemplos do uso de extensões já desenvolvidas, que foram usadas para acelerar a parametrização de aplicações SCADA:

a. Como vários usuários utilizavam com frequência um determidado CLP (poderia ser uma UTR, um IED, etc.), desenvolveu-se uma extensão que lê o arquivo de configuração deste CLP e gera, automaticamente, os templates com os tags das variáveis definidas no IED, criando uma regra de nomeação para os mesmos. Isso foi feito com o CLP ControlLogisx da Allen-Bradley.

b. Para substituir aplicações SCADA antigas pela nova, desenvolveu-se um importador que lê tanto a base de dados do SCADA antigo como também as telas existentes e gera uma aplicação no novo software, com garantia de que todos os pontos já comissionados serão gerados corretamente, já que os endereços são copiados.

c. Como se usa com frequência os protocolos orientados a objetos IEC-61850 e OPC, desenvolveu-se duas extensões que lêem os IEDs IEC-61850 e servidores OPC, respectivamente, e geram a base de dados de tempo real. No caso do IEC-61850, pode ser usado também o arquivo SCD como fomte para importar os dados.

d. Como determinados usuários usavam muito a base de dados PI, da OsiSoft, desenvolveu-se uma extensão que importa os dados desta base em tempo de projeto e permite, em tempo real, usar estes dados para gerar relatórios e Dashboards.

e. No uso de um algorítmo FLISR (fault location, isolation, system restoration) em uma aplicação SCADA, existe um primeiro programa que gera uma rede de distribuição com vários laços que serão monitorados pelo FLISR e popula-se seus dados a partir de dados do sistema GIS da concessionária. Ao final do programa, tem-se todos os dados em uma base de dados relacional que possui, não só a rede distribuída que será controlada, como as telas representado os laços. Desenvolveu-se uma extensão que não só importa todos os dados da rede de distribuição para o SCADA, como gera as telas com os laços que serão controlados [4].

f. Se o usuário sempre usa uma mesma planilha, com um formato padrão, para  para entrar com os dados de todos seus tags, com endereços, limites operacionais, eventos, alarmes, etc., pode desenvolver uma extensão que lê essa planilha e gera sua base de dados, eventos, alarmes, histórico, etc.

g. No caso da metodologia Lean Automation, gerou-se uma extensão que a patir dos templates definidos gera, automaticamente, todos os alarmes/eventos da aplicação, assim como as condições de gravar registros históricos e a lista de endereços de todos os pontos.

Generalizando, com a funcionalidade extension, sempre que existe um padrão utilizado muitas vezes para gerar a base de dados do SCADA, é possível desenvolver uma extensão que automatiza este processo, minimizando a possibilidade de erros e reduzindo o tempo de configuração e teste.

## 5.0 CONCLUSÃO

Neste trabalho foi mostrado uma nova tendência aplicada aos softwares tipo SCADA que é disponibilizar um conjunto de ferramentas no software que permitem criar componentes pré-prontos que, tal qual um jogo de Lego, permitem automatizar o processo de geração de aplicações SCADA, conectando-se componentes diversos. Além disso a metodologia fixa as tecnologias de uso dos IEDs (melhores práticas) ao software, permitindo que a mesma fique na empresa e não na cabeça de seus técnicos.
Na sequência do desenvolvimentoi da tecnologia, o próximo passo será a distribuição de bibliotecas de

componentes que permitirá que usuários construam e testem novas aplicações em horas ao invés vários dias.
Além da automação da geração de aplicações, é possível adicionar-se à solução, bibliotecas de terceiros, sejam elas “open source” ou não e, com o uso dos componentes, moldar essa biblioteca para funcionar dentro de determinados limites, retirando do usuário o tempo e a obrigação de estudar a biblioteca externa de maneira a usa-la, isto é, o componente disponibiliza os métodos para utilizar a biblioteca como se a mesma fosse uma funcionalidade do software SCADA.
Concluíndo o processo, adicionou-se uma última ferramenta designada extensão que permite automatizar a importação de dados oriundos de fontes muito usadas por um conjunto de usuários do software.
O resultado final é o aumento da produtividade dos integradores de sistemas SCADA com a redução de custo e tempo de projeto.

## 6.0 BIBLIOGRAFIA

[1] Rampazzo, W.A, Teixeira, R.C, e Amorim, F.G.A, “Solução Descentralizada para Digitalização de Subestações Utilizando o Protocolo DNP3.0”, XVII SENDI, Olinda, jan/2015.
[2] Silva. C.A, Silva Junior.A.P e Borges. R.L. “Desenvolvimento de Sistema Especialista para Gerenciamento de Transformadores instalados em subestações”, Cigré-Brasil, “IX Workspot - Workshop Internacional Sobre Transformadores de Potência, Equipamentos, Subestações e Materiais, 2018.
[3] Silva, T. P, Simões, C. “Integração de Ferramentas Open Source em Software SCADA – Um atalho em direção ao Smart Grid”, Cigrè Latino Americano, XVIII ERIAC, Foz do Iguaçu, Maio 2019.
[4] Simões, C., Porto, J.A.S.B., Kagan, H. Pelegrini, M.A. e Guaraldo J.C. “Implementação de Algoritmo de FLISR”, Cigrè Latino Americano, XVII ERIAC, Foz do Iguaçu, Maio 2017.

## 7.0 DADOS BIBLIOGRAFICOS

- Nome: Clovis Simões
- Local e ano de nascimento: Curitiba - PR, 13/11/1951
- Graduação e pós-graduação: Graduado em engenharia mecânica, UFRGS, 1973 e MSC em Ciência da Computação, UFRGS 1977.
- Experiência Profissional: É sócio fundador da Spin Engenharia de Automação, tendo participado no desenvolvimento de quatro sistemas SCADA, sendo dois já na Spin Engenharia (ActionView [EMS/SCADA] e Action.NET [ADMS/SCADA]). Participou em dezenas de projetos de automação de usinas, subestações, complexos eólicos e centros de controle de geração e distribuição. Já escreveu e apresentou dezenas de papers, em congressos nacionais e internacionais, sobre desenvolvimento de software e automação de sistemas elétricos, máquina de papel e controle da propulsão e avarias de navios. Foi professor na Universidade de Brasília e UFRGS, na área de ciência da computação.